UNIVERSITY OF OSLO
Department of Informatics

Multi-threaded conversations in instant messaging clients

Master thesis
60 credits

Are Wold

May 2, 2007

Abstract

Instant messaging is seeing increasingly widespread use in both leisure and work contexts. Several studies have illustrated problems associated with the textual chat interfaces typically used in instant messaging clients.

In this thesis, we prototyped alternative interfaces for conducting instant messaging conversations, attempting to create an interface which would increase the perceived conversation quality in multi-threaded conversations. The final prototype, TopicMessenger, let the users divide the conversation into explict topics by placing messages in separate panes in the user interface. We believed this functionality could reduce problems associated with multi-threading and make conversations more coherent, while remaining easy to use.

TopicMessenger was compared to StandardMessenger, a version of the software without any special features, in an experiment with 35 participants. The participants in the experiment were students aged from 19 to 30. The conversations in the experiment were placed in a non-professional context, and participants discussed leisure topics ranging from travel to music. Both two-way and four-way conversations were conducted. In a conversation, different users were given different topics to focus on, in order to provoke multi-threaded conversation.

Through log analysis, questionnaires and open-ended group interviews, we found that the conversations in TopicMessenger had less multi-threading and users felt they were interrupted less often while using TopicMessenger. Four-way conversations using TopicMessenger were perceived as more coherent than four-way conversations in StandardMesenger. Users described the topic functionality as easy to use and useful, and expressed an intention to make use of topic functionality if it is made available to them.

In conclusion, we consider the usefulness of explicit topics in a more structured context, and argue that topic functionality would be a useful addition to commercial instant messaging clients.

Table of Contents

Index of Tables

Index of Figures

Preface

This thesis is the conclusion of a two-year master's degree in Informatics at the Department of Informatics at the University of Oslo. The thesis itself is worth 60 ECTS credits, equivalent to one year of studies.

I would like to thank my supervisor, Senior Scientist Erik G. Nilsson at SINTEF ICT, for his very valuable guidance and support during the work with this thesis. Through Erik, I have also been fortunate enough to receive ideas, feedback and suggestions from other SINTEF scientists. In particular, I would like to mention Asbjørn Følstad, whose assistance in shaping the experiment and providing statistical advice has been vital. Jan Håvard Skjetne and Odd-Wiking Rahlff have offered useful ideas and suggestions.

I would also like to thank my internal supervisor, Steinar Kristoffersen at the Department of Informatics, for his assistance.

Finally, I would like to express my gratitude for the support I have received from my girlfriend, Margrethe Store. Thank you!

1 Introduction

By the term “instant messaging”, we refer to the exchange of typed text messages sent over the Internet. Two or more users participate in a conversation, and messages typed and sent by one user are visible almost instantaneously to the other users.

Compared to other communications media commonly used today, such as telephone and e-mail, instant messaging occupies a middle ground. It is not completely synchronous, like a telephone call or face to face conversation is, yet it is not completely asynchronous, like an exchange of e-mails. While there is no “conversation transcript” available to the participants in an oral conversation, participants in instant messaging conversation have access to the previously received messages via the chat history.

The special characteristics of instant messaging can result in misunderstandings and confusion during conversations. In this thesis, we attempt to create an instant messenger design that reduces these problems.

Many of today's instant messaging clients offer a host of features such as file transfer, the ability to see which of the user's contacts who are online, what music they are listening to and more. Here, we focus exclusively on the conversational aspect of instant messaging.

1.1 Motivation

As the Internet has become more and more widespread, the adoption of instant messaging has increased, both for leisure and work purposes. In 2005, 70% of American Internet users used instant messaging, and 26% of employed instant messaging users said they used instant messaging in the workplace (America Online 2005). In 2004, the Pew Internet and American Life project reported that the American adult instant messaging population had increased from 41 million in 2000 to 53 million (Pew / Internet 2004). ComScore found that 82 million people used instant messaging in Europe in February 2006, accounting for about half of the Internet users (ComScore 2006).

It is clear that instant messaging is becoming an important tool, both in people's everyday lives and in business. Interestingly, the instant messaging chat interface remains basically the same as the chat interface used around 1990.

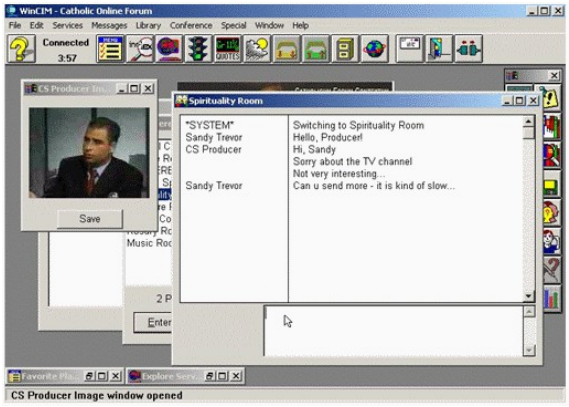


Figure 1.1: CompuServe CB (Living Internet c), ca. 1990. Chat window to the right.

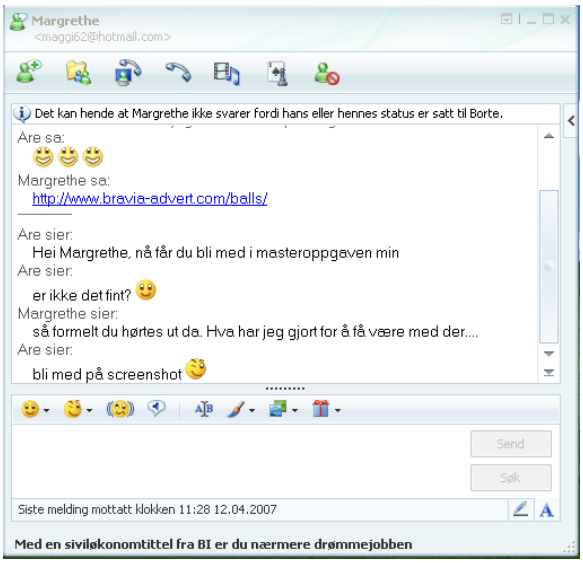


Figure 1.2: Chat window from Windows Live Messenger, April 2007.

The main interface in Windows Live Messenger[1] (Figure 1.2) today is similar to that of WinCIM from around 1990 (Figure 1.1). The user types a message into a text entry pane, and when the message is sent, it is placed into the conversation pane immediately above. Messages are ordered chronologically, with the most recent messages at the bottom.

Several studies have shown problems arising from this method of structuring conversations. Conversation participants discuss several subjects at the same time, and as a consequence, messages that relate to different topics are placed next to each other in the chat history. Different users can compose messages simultaneously, resulting in disjointed conversations. Garcia and Jacobs (1998) show how users suffer misunderstandings because of messages that appear to be related, but are in fact not. The misunderstandings make the conversation inefficient and hard to follow. Voida et al. (2002) show users suffering from similar confusion and frustration as they conduct conversations where several threads of conversation occur simultaneously.

A more recent study, by O'Neill and Martin (2003), shows how chat users at a series of online business seminars cope well with multi-threaded conversations.

In this thesis, we explore alternative interfaces for text chat, with the aim of improving the user experience and conversation quality. Also, we want to investigate whether the inherent problems of text chat as described by Voida et al. and Garcia and Jacobs are causing significant confusion, or if they are handled routinely, as described by O'Neill and Martin.

Smith et al. (2000) attempted to improve on the traditional chat interface, and tested their prototype with structured, work-related tasks. We focus on less structured conversations as we believe these are more common in the everyday use of instant messaging.

[1] Windows Live Messenger is the latest version of the software from Microsoft previously known as MSN Messenger. Windows Live Messenger has also partly supplanted another Microsoft messaging product, Windows Messenger (Wikipedia 2007g).

1.2 Method overview

In this thesis, we used a mixture of qualitative and quantitative methods, with an emphasis on the quantitative methods. By using different data gathering methods, we aimed to triangulate the results and thereby improve our understanding and aid our interpretation of them (Mathison 1988).

1.2.1 Literature review

We conducted a literature review to summarize relevant existing books, articles and other documentation on the subject matter. In this thesis, we focused primarily on literature related to different categories of online textual communication. As we place the emphasis on quantitative methods, we attempt to use the literature review as a basis for our research questions and hypotheses (Creswell 2003).

1.2.2 Prototyping

Prototyping is regarded as an important aspect of the design process, and can help the designer test the technical viability of an idea and conduct user testing and evaluation (Preece et al. 2002). In this thesis, we used the prototype to investigate different design ideas for instant messenger applications by creating different prototypes and developing them iteratively. Finally, we conducted an experiment in which participants used two prototypes. One, StandardMessenger, was a baseline prototype, which had only basic text chat functionality, while the other, TopicMessenger was enhanced with new functionality.

1.2.3 Experiment and statistical analysis

We conducted an experiment where we had participants try two different instant messaging prototypes in a total of four conversations, in other words a within-subjects repeated measures experiment (Creswell 2003, Greenwald 1976).

From the experiment, we gathered both log data and questionnaire responses. The log data was submitted to manual and automatic analysis to gather statistics on the conversations. The questionnaire data was used to create statistics on the participants' perceptions of the conversations and collate information about the participants themselves.

Finally, we used descriptive statistics as well as Student's t-test (Moore and McCabe 2003) to compare participants' perceptions of the conversations in the different prototypes and the data gathered from the log analysis of the conversations. The use of statistical tests is further discussed in section 5.8, Statistical analysis.

1.2.4 Open-ended group interviews

At the end of the experiment, we conducted an open-ended group interview. One of the benefits of qualitative research as described by Creswell (2003) is the holistic perspective it can help establish. Through the interviews, we wanted to gain a more complete understanding of the participants' user experience than we could get from questionnaires and log data alone.

1.3 Outline of this thesis

Chapter 1 is this introduction.

Chapter 2 discusses the history of instant messaging and various uses it is seeing today. It then proceeds to give an overview of the main problems with instant messaging conversations and possible solutions as described in the existing literature.

Chapter 3 describes the design process, in which we attempt to create a prototype offering improved conversation quality compared to typical instant messaging clients.

Chapter 4 sums up the research problems underlying the thesis and details the hypotheses designed to examine them.

Chapter 5 gives a detailed overview of the design of the experiment conducted to test the hypotheses.

Chapter 6 gives the results from the experiment.

Chapter 7 discusses the results from the experiment and the validity of the experiment design.

Chapter 8 outlines possible avenues for future research.

Chapter 9 is the conclusion of the thesis.

2 Background

In this chapter, we define instant messaging, give an overview of its workings and history, and summarize the most relevant research on the topic. Finally, we highlight the problematic facets of instant messaging relevant to this thesis and possible solutions.

2.1 What is instant messaging

By instant messaging (IM), we refer to the exchange of text between two or more users across a network using instant messenger software. The software is usually but not exclusively run on personal computers. Key features of IM are the "buddy list" – a list of people using a compatible IM client that displays their "online status" (whether or not they are available for chat) and the immediacy of the communication. A message typed in by one user will usually be visible to other users within a negligible amount of time. IM shares many characteristics with face to face communication and is often compared to it (Voida et al. 2002, Garcia and Jacobs 1998).

2.1.1 A brief history of instant messaging

One of the roots of instant messaging can be found in the utility "talk" (Nardi et al. 2000), which became available in Unix in 1973 (Greene et al. 2004). "Talk" permitted a user on a computer terminal logged in to some machine to send text to another logged-in user on the same machine (Living Internet a). The two users would be logged in on the same server, but via separate terminals. In "talk", every character typed by the user is instantly transmitted to the other user, in contrast with most current instant messaging and chat applications, which transmit statements – a sequence of characters followed by the Enter keypress.

Talkomatic, whose development started in 1973, and the derived term-talk are other examples of early chat and instant messaging systems. While Talkomatic enabled group chats with up to five participants, term-talk's purpose was one on one-communication. Term-talk could be used without exiting the program the user was currently running, similar to an instant messaging service (Living Internet b).

Online text chat went public with CompuServe's CB program, in which multiple users could talk to each other (Wikipedia 2007b). At its peak in 1988, it had more than a million subscribers (Living Internet c).

In the summer of 1988, the Finnish graduate student Jarkko Oikarinen began creating Internet Relay Chat (IRC). He was inspired by other early chat programs, such as Bitnet Relay Chat and rmsg (Oikarinen). IRC spread rapidly, and today it is still seeing heavy usage. Two of the most popular networks (a collection of servers whose users can all talk to each other), EFNet and Undernet, have as of this writing about 65.000 and 120.000 users, respectively (EFnet 2007, Undernet 2007). An example of a modern IRC client is mIRC[2] (Figure 2.1).

[2]mIRC: http://www.mirc.com (Retrieved April 16, 2007)

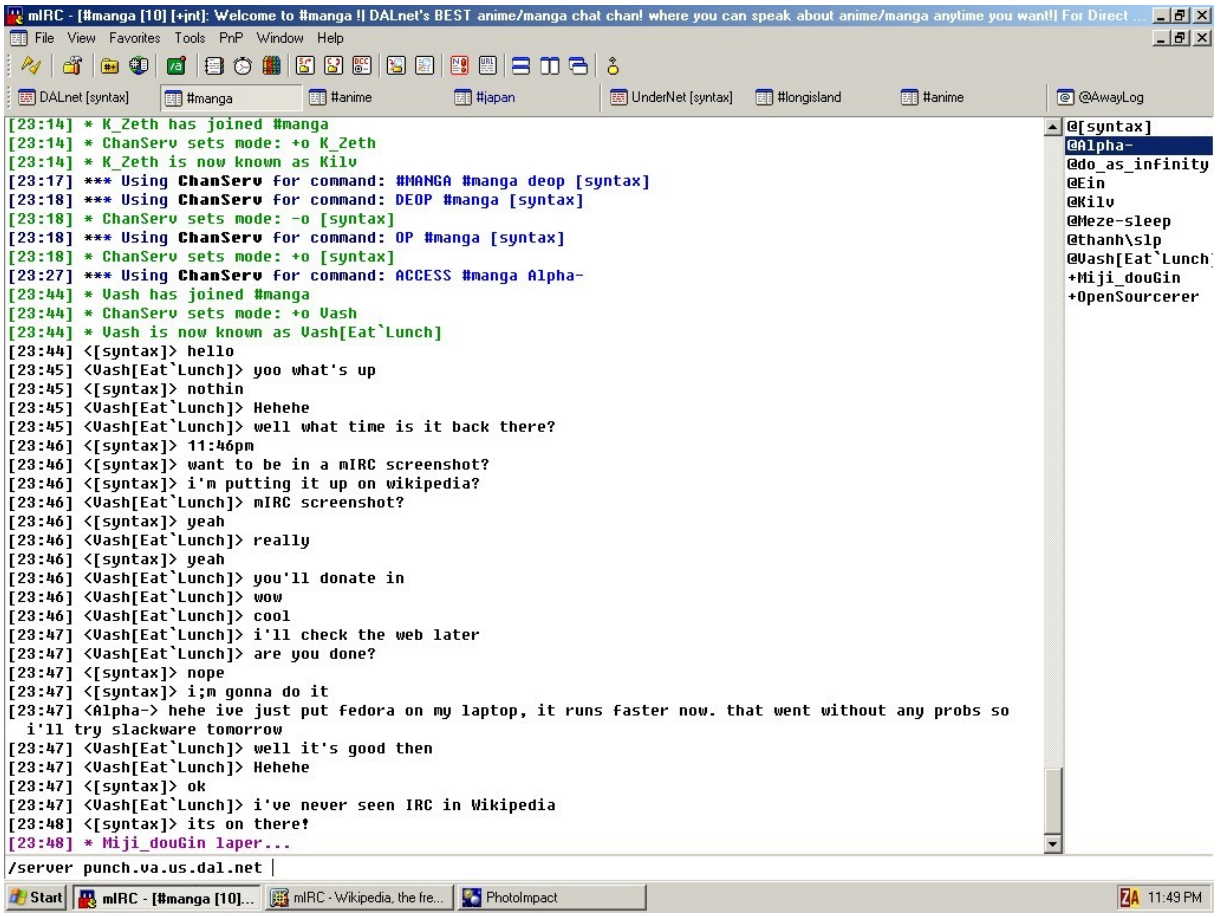


Figure 2.1: mIRC – a modern, graphical IRC client (Wikipedia 2007i).

The first "modern" instant messaging client, ICQ, was released on November 15. 1996 by Mirabilis[3]. It was created by four young computer enthusiasts. Adoption rate was fast – after six weeks, 30.000 people had tried it, after six months, a million. In 1998, the number of ICQ client downloads had reached 16 million, and at the end of 2001, the count was 117 million (Kaufman and Li 2003).

ICQ was not alone in the market for long. America Online released the AOL Instant Messenger in 1997, Yahoo! released their Messenger in 1998, and Microsoft their MSN Messenger in 1999 (Wikipedia 2007a, Wikipedia 2007e, Wikipedia 2007d). In 2004, Pew Internet & American Life Project estimated that these services had 85 million active users in total, "active" meaning they had logged on to their accounts within the last month (Pew / Internet 2004).

In the US in 2005, 70% of Internet users used a form of instant messaging and 38% sent more instant messages than e-mails (America Online 2005). In 2005, an analyst estimated that the number of Norwegian IM users reached 1.2 million, roughly 25% of the population. One billion instant messages were sent in Norway per year, and traffic increased by 32% during the second half of 2005 (DagensIT 2006).

[3] Mirabilis, a small Israeli firm founded in 1996, was acquired by AOL in June 1998, and is today a part of Time Warner alongside AOL (Wikipedia 2007h).

2.1.2 Instant messaging infrastructure

Instant messaging is usually a client-server system. A user has a set of "buddies" - people (s)he knows that also use the instant messaging service – and when the user connects to the server using the client application, the server notifies all his/her buddies that (s)he is connected, and provides the user with a list of all his/her buddies and their status.

The user can then send messages to his/her buddies. Depending on the instant messaging network, these will travel to the server, which forwards them to the recipient, or they are sent directly to the recipient from the client application. In the latter case, the server will have distributed the buddies' IP addresses to the user (Gowan 2000).

There are also examples of decentralized instant messaging systems, such as Dawn (Greene et al. 2004) and CSpace[4]. These are not dependent on a central server to exchange messages and presence information between its users.

Instant messaging is dominated by proprietary networks, such as those operated by Microsoft, Yahoo! and AOL, respectively (Pew / Internet 2004). Historically, users of one proprietary network could not communicate with users of another such network without using some sort of intermediary software. This situation is starting to change with the introduction of interoperability between Yahoo! and Microsoft messaging software (Microsoft 2006). There are also multi-network clients, such as Pidgin[5], that permit the user to connect to different networks with the same application. However, these are typically not full-featured when compared with the proprietary clients (Wikipedia 2007c).

2.1.3 Instant messaging and text chat in other arenas

Text chat formed an important part of multi-player games already in the late seventies, with the advent of the first Multi-User Dungeons (Bartle 1990). A Multi-User Dungeon (MUD) is a text-based virtual reality space where several users can log on at the same time, move around and interact with each other (Curtis 1992). Today, text chat is the primary means of interaction between players in most massively multiplayer online games, the descendants of MUDs (Ducheneaut and Moore 2004).

Other uses of text chat can be found in music sharing applications such as Aimster (Grinter and Palen 2002), web conferencing tools (IBM Lotus Sametime 2007) and online collaboration tools such as Google Docs & Spreadsheets[6], among other applications. In these contexts, the text chat facilitates communication between the people sharing music with each other, watching a presentation together or composing a document together.

Interestingly, while text chat is being embedded into these applications, there is also a trend towards the messaging client taking on more and more responsibilities. Windows Live Messenger handles video and voice calls in addition to the traditional chat and file transfer typically seen in instant messaging clients. Multi-player games, remote desktop control, a whiteboard and application sharing are also available (Wikipedia 2007f). The Internet

---

4 CSpace: http://cspace.in (Retrieved April 16, 2007)

5 Pidgin, a multi-protocol instant messaging client formerly known as Gaim: http://www.pidgin.im/ (Retrieved April 16, 2007)

6 Google Docs & Spreadsheets: http://docs.google.com/ (Retrieved April 16, 2007)

telephony program Skype[7] has the "standard" chat functionality, as seen in clients from Microsoft and Yahoo!, but focuses on video calls, text messages to mobile phones and telephony to other Skype users as well as ordinary phones around the globe.

As more and more applications use text chat for collaboration and the messaging clients gain more functionality, we note that the core text chat interface and functionality remains fundamentally unchanged (Smith et al. 2000). In this thesis, we concentrate only on this core functionality and potential improvements to it.

2.1.4 "Instant messaging" versus "text chat"

"Text chat" refers to near-synchronous textual communication between several users, as it is used by O'Neill and Martin (2003) and Smith et al. (2000) among others. "Instant messaging" refers to the "exchange of typed messages between computer users in real time via the Internet" (McKean 2005). These definitions are overlapping.

Grinter and Palen (2002) see instant messaging as the most recent and popular example of text chat communications, but one which, in contrast with IRC and MUDs, has a larger emphasis on communication between people who know each other rather than communication between strangers.

Communication via instant messaging also tends to have a smaller amount of users in the conversation, for instance, Isaacs et al. (2002a) studied IM use with a client that only permitted one-on-one conversations. Nardi et al. (2000) focused on the "more typical" two-way conversations instead of group conversations, which were not supported by all clients. In contrast, Hentschel (1998) and Schulze (1998) discuss communication on IRC, showing log excerpts from group conversations with four or more participants.

Grinter and Palen (2002) hypothesize that teenage users of instant messaging see it more as a general communications tool, while IRC and MUDs are conceptualized as destinations where they can meet like-minded people.

In this thesis, we focus on two and four-way text chat conversations of the sort that friends would conduct with friends. Considering the large number of users on the mainstream instant messaging networks versus the number of users on the IRC networks, we believe that the primary arena for this type of text chat conversation today is instant messaging clients. Therefore, we use the terms "text chat" and "instant messaging" interchangeably.

2.1.5 Presence awareness

A major feature of instant messaging is the presence awareness enabled by the "buddy list". A user can see which buddies are online, and, depending on the IM system, if they are busy or available for chat, and even what music they are listening to. Presence awareness is not the subject of this thesis, but can be further explored in Campbell et al. (2003), Erickson et al. (1999), Herbsleb et al. (2002) and Chatterjee et al. (2005), among others.

[7]Skype: http://www.skype.com (Retrieved April 16, 2007)

2.2 Overview of relevant chat and IM research

In this section, we give a chronological overview of recent research in instant messaging conversations, using a table similar to that used by Cameron and Webster (2004).

Source	Focus	IM system	P.	Methods	Findings	Relevance
Shen et al. (2006)	Automatic thread detection	Text chat		Statistical evaluation of algorithm	Significant improvement over earlier algorithms	Alternative way to handle threads
Lonchamp (2005)	Making chat an effective learning tool	Custom (Omega Chat)	-	Open source distribution	Created malleable chat system	Let the user decide how much structure / change is desired
Tran et al. (2005)	Is awareness support sufficient in current clients?	Custom	173	Survey	Current clients provide too little awareness	Multi-conversations are common
Ogura et al. (2004)	Automatic log analysis for knowledge creation	Analyzed log data from 2-3 way chat	21	Log analysis	Algorithm sorts about half of statements into correct category	Automatic method to classify statements into threads
Zitzen and Stein (2003)	Is chat similar to oral communication?	Analyzed log data from IRC	52	Log analysis	Chat medium must be defined in its own right	Second pair in adjacency pairs often several lines apart
Campbell et al. (2003)	What happens during message composition?	Custom	16	Experiment	Lack of awareness could cause inefficient conversations, only 43% of time spent typing	Typing activity not indicative of attention to IM
Grinter and Palen (2002)	How do teens use IM?	MSN, Yahoo!, ICQ, AOL	16	Interviews, online observations	Teen IM use charcterized by social factors and communications media choice	Users multitask, keeping track of many conversations difficult
Voida et al. (2002)	Tensions between IM-chat and verbal communication	Standard software	8	Experiment	Analysis of communicative conventions can prevent design tensions	Users might not track multiple threads; interrupt each other
Isaacs et al. (2002b)	How is IM used in the workplace?	Custom software (Hubbub)	437	Experiment	Primary use of workplace IM is complex discussions	Heavy users thread their conversations
Smith et al. (2000)	Can chat ambiguity be avoided with a better client?	Custom (Threaded Chat)	70	Experiment	Users' pattern of interaction is equally effective in standard chat and “Threaded Chat”	Users adapted to new interface, significant UI challenges remain
Viegas et al. (1999)	Visualize social structure of conversation	Custom (Chat Circles)	-	Informal trials	Visualization of online conversations in 2D space	Potential way of handling multi-threading
Erickson et al. (1999)	Support computer-mediated communication (CMC) through persistence and social proxies	Loops and Babble	50+	Long-term trial	Informal, persistent communications systems fill a niche	Need tools for structuring conversation
Herring (1999)	Coherence in CMC	IRC, MUD		Research survey	CMC has both interactional advantagas and disadvantages,	Mechanisms for two-way message production and linking of related statements
Vronay et al. (1999)	See chat as streaming media type. Provide more information about other users' activities	Status Chat, Flow Chat	10, 30	Experiment	Radically different interface introduced new usability problems	Too much activity and novel interface means distraction
Garcia and Jacobs (1998)	Consequences of importing oral procedures to chat	Custom (Aspects)	3	Experiment	Importing procedures from oral conversation into computer-mediated conversation resulted in misunderstandings and confusion	“Phantom responsiveness”, “Phantom adjacency pairs”, “Virtual simultaneity”

Table 2.1: Chronological overview of recent research in chat and instant messaging.

2.3 Problems with instant messaging conversations

2.3.1 Definitions

Adjacency pair: A pair of statements where the "occurrence of the 'first part' of a pair defines the occurrence of the 'second part'" (Warren 2006, p141). Typical examples are greeting/greeting, question/reply and offer/acceptance.

Conversation: A speech event outside of an institutionalised setting involving at least two participants who share responsibility for the progress and outcome of an impromptu and unmarked verbal encounter consisting of more than a ritualized exchange (Warren 2006).

Instant message, message: A sequence of typed symbols composed in the text entry pane of an instant messaging application. The message is composed and then sent; the receiver does not see anything of the message contents until it is explicitly sent by the sender.

Interruption (thread): Interruptions within a thread occurs when there is no relation from a statement to the previous one, but the statement is still on the same topic as the previous statement. This pheomenon is termed "topic drift" by Warren (2006, p.166.). For instance, Anna might interrupt George:

George:	I can't really say I liked this movie.. what did you think about the special effects?
Anna:	Hmmm... this wasn't the best acting I've seen this year, but certainly not the worst either

Interruption (topic): A topic interruption occurs when a statement is related to a different topic altogether than the statement preceding it. Warren (2006, p.173) refers to this phenomenon as "topic shift".

George:	I can't really say I liked this movie.. what did you think about the special effects?
Anna:	How about getting a new pair of skis for Lisa?

Phantom adjacency pair: Two statements appearing one after the other in the chat pane where the last statement appear to be a reply to the first. The last statement is, however, a reply to a different statement (Garcia and Jacobs 1998). In the following situation, several factors could cause George's late reply – maybe he is a slow typist compared to Anna, or maybe he was paying attention to a different program when Anna sent her messages.

Anna:	The movie was pretty awesome
Anna:	Those special effects were mindblowing
Anna:	But I have to say I thought the dinner afterwards was even better :)
George:	Yeah, it was really nice, I enjoyed it :)

Repair statement: A repair statement is a message created solely to correct an apparent misunderstanding caused by mis-placed turns.

Thread: A conversation thread is a sequence of turns that are related to one another (Smith et al. 2000).

Turn: In oral conversations, participants generally speak in turns, one at a time (Warren, 2006). A turn can consist of an incomplete utterance or several utterances from the same participant. A message is the text chat equivalent of a turn.

Utterance: An uninterrupted chain of spoken or written language (McKean 2005). In instant messaging, one utterance can be composed of several instant messages. Consequently, an utterance can be broken up when another conversation participant inserts a message between two messages belonging to the same utterance.

2.3.2 The quasi-synchronous nature of instant messaging

Instant messaging converations are influenced by both written and spoken types of interaction. While the medium is in written (typed) form, the conversations that take place have much in common with spoken conversation (Voida et al. 2002).

Garcia and Jacobs (1998) coined the term quasi-synchronous to describe the nature of instant messaging. While programs such as *talk* on Unix transmits every keypress the user makes, replicating some of the immidiacy of spoken interaction and its synchronous nature, instant messaging decouples message production and message consumption. The message is not sent until the user presses the "Enter" key; the recipient gets all of the message at once.

Voida et al. (2002) refers to the problems related to the quasi-synchronous nature of instant messaging as synchronity tensions, and points out that users exhibited a desire to make the communication feel asynchronous. They were not willing to wait for the chat participant's reply before writing more themselves. This resulted in multi-threaded conversations, which were sometimes confusing.

2.3.3 Turn-taking

The decoupling of production and consumption of turns has ramifications for turn-taking. In spoken conversation, a *turn* is one participant's contribution in the conversation before the turn moves to someone else. Warren (2006) describes three methods of choosing the next speaker in a conversation:

Designation, in which the current speaker selects the next speaker explicitly, for instance by asking a question to someone.

Constriction, in which the speaker constrains the topic, so that a certain reply is expected. Someone then has to provide this reply, when they might otherwise have their own topic to pursue in the conversation.

Self-selection, in which participants in the conversation have to recognize "Transition Relevance Phases", or TRPs. When conducting an oral conversation, it is obviously possible to monitor the current speaker, both his/her gestures and his/her words, and thus make a qualified judgement as to when it is a suitable time to contribute to the conversation.

Voida et al. (2002) point out that there is no communicative artifact that structures turn-taking in IM conversation, because participants are able to contribute to the conversation at the same time. It was difficult to judge from the conversation transcript whether a series of messages were intended to be a complete utterance, which would mean that other conversation participants was free to contribute, or if more messages were coming.

Voida et al. (2002) gives an example of turn-taking confusion:

Jen: Sigh. . . no more news on Donna from dad.
Grace: Have you heard any more from your dad? I do not have any mail fom Diane. You just answered my question.
Grace: Do you know a game called Spider. It is a type of Solitaire. Laura says she likes it.

Jen: Go get your shower and get to bed... I hope your stomach calms down soon.
Jen: Haven't heard of Spider.
Grace: Thank you. I will be anxious to hear from you tomorrow. Good night and thanks again for the dinner.

The participants in the conversation were typing simultaneously, but there is no mechanism for signifying whether they are done "speaking" or not, and no indicator telling their conversation partner how much work they have done on their turn. None of the three types of turn control mechanism are in play. As a result, Grace's ouverture about the game Spider is interrupted by Jen's comment about going to bed, and Grace is prompted to end the conversation (Voida et al. 2002).

Some messaging systems, like Microsoft's Windows Live Messenger, offer an indicator telling the participant whether the other side(s) in the conversation are typing at the moment, which helps signify that the coast is clear for the next would-be speaker. However, the usefulness of this feature for the purpose of turn-taking control is limited, due to the instant and short messages typical of instant messenger conversations (Tran et al, 2005; Dix et al., 2004) and the fact that the contents of the message remains unknown until it is complete and sent (Smith et al. 2000).

2.3.4 Multi-threading and phantom adjacency pairs

Since users can compose messages without interrupting each other (unlike in oral communication) and since there is a lack of turn-taking control, there is a tendency to create more than one thread of conversation (Voida et al. 2002). In conversations between slow and fast typists, the fast typist might ask a question, and if a reply does not appear as quickly as expected, send another question before the other side can send the reply to the first (Vronay et al. 1999, Voida et al. 2002).

Jacobs and Garcia (1998) show that when statements from different threads are intermingled, "phantom adjacency pairs" can ensue and cause confusion. Phantom adjacency pairs are messages that appear to have a relation to each other, but in fact are related to messages not directly adjacent to them. From Garcia and Jacobs (1998):

108 Silver: AND YOU ARE ANGRY THAT YOUR CREDITS DID NOT
109 FOLLOW YOU
110 Mr White: IS THAT IT?
111 FRED: YES I AM ANGRY THAT I LOST A LOT OF CREDITS BUT
112 THATS WHERE I FELL INTO A HOLE WITH THE FIRST FINAL OF THIS
113 PAPER I TENDED TO FOCUS IN ON THAT ANGER AND NOT ON THE
114 LOGICALITY OF THE BIG PICTURE
115 Mr White: AH HA
116 Mr White: AND NOW YOU WILL CHOOSE TO FOCUS ON THE LOGIC,
117 INSTEAD OF THE EMOTION ...
118 Silver: OAKY SO YOU NEED TO BE DETACHED FROM YOUR
119 FEELINGS WHEN WRITING THIS ESSAY... EASIER SAID THAN DONE
120 FRED: NO THATS NOT IT THEY DO HAVE ALSORTS OF CRACKS IN
121 THE SYSTEM THAT IF YOU KNOW HOW TO PLAY THEM YOU CAN GET
122 MORE CREDITS TO TRANSFER THAN YOU COULD IF YU DID'NT KNOW HOW
123 TO DODGE THE SYSTEM

In this log, there are several apparent adjacency pairs. Lines 111 – 114 appear to be a reply to line 110. Lines 120 – 123 appear to be a reply to lines 118 – 119. When examining the video log, it was in fact discovered that line 110 didn't exist when lines 111 – 112 were composed,

and that the same is true for lines 120 – 121. Fred's first statement was a reply to line 108, while the second is a reply to lines 110. There are no explicit links between the related messages, and this can be a source of misunderstandings.

Voida et al. (2002) show another example of the confusion multi-threadedness can cause:

Eric: Kitties don't like traveling in airplanes
Kate: they let you bring them on the plane
Eric: (Well, for that matter, neither do fish)
Katie: no?
Katie: they still alive?
Eric: One is
Eric: But I want to see you teach a cat how to pop its ears
Eric: That would warrant a Nobel prize, at the very least
Katie: true true
Katie: I want this cat at the store...it's like two years old, but it's the coolest cat ever
Eric: Cool how?
Katie: Totally friendly . . . ready to cuddle and love ya
Katie: if it's still there in a couple weeks I'm gonna see about getting it
Eric: Cool
Katie: ya know, animals can pop their ears
Katie: cats, dogs, hamsters
Eric: Huh

The conversation involves two threads; when Katie returns to the first thread at the end of the transcript, Eric appears to give up (Voida et al. 2002).

A consequence of this intermingling of different sequences of messages (threads), is that people often have to scroll through the conversation transcript to find the previous message in the sequence, which requires effort and is time consuming (Smith et al. 2000).

If threading was rare, the effort expended on "repairing" misunderstandings caused by multi-threading would be relatively small. However, Isaacs et al. (2002b) report that threading is common between "heavy" instant messaging users. "Heavy users" was defined as users with more than three conversations per day. In this group, 41% had multi-threaded conversations. Voida et al. (2002) report that threading is "extremely commonplace".

2.3.5 Virtual simultaneity

Related to the problem of phantom adjacency pairs and multi-threading, is that of "virtual simultaneity" (Garcia and Jacobs, 1998). While oral conversations often overlap (Warren 2006), instant messages cannot – they are always arranged chronologically by the time they arrive at the server (or client, depending on the IM system). This results in virtual simultaneity. If a user is paying attention to the instant messaging client, (s)he will know that two messages that arrive nearly at the same time are independent of one another. However, if his/her attention is diverted for a moment, the second message can appear to be a reply to the first – even though they were composed at the same time.

2.4 Solutions to problems of instant messaging conversations

2.4.1 Is a solution needed?

O'Neill and Martin (2003) report that users manage the complexity of multi-threaded chats with relative ease. In their paper, they focus on the practical aspects of instant message conversation – how participants take turns, how they clear up misunderstandings, and so on.

The participants in the study are academics and businesspeople taking part in online discussions after online business seminars. They have varying levels of experience with instant messaging tools (the study does not measure their level of experience in any way), and most of them have not used the specific communications tool available before.

O'Neill and Martin write that it is precisely the quasi-synchronous nature of chat that makes it possible for the participants to cope with the multi-threaded nature of the conversations. Since only complete comments are posted, it is possible to read the log and piece together the threads in the intervals between posts. Participants' contributions to a thread can be connected to the previous entry in the thread, even if there are several messages in between. When participants had multiple threads to reply to, they did so in separate messages, preserving the threads.

The paper references Garcia and Jacobs' (1998) phantom adjacency pairs and remarks that they rarely occurred. When a participant became confused as a consequence of phantom adjacency pairs, the confusion was quickly cleared up by the other participants in the chat.

Here, we would like to point out that the risk of misunderstandings caused by phantom adjacency pairs is larger when participants aren't devoting all of their attention to the chat – which it seems they were doing at the chat events described in this study.

Another point worth noting is that the participants comment multiple times on how the chat is developing in the log excerpts provided by O'Neill and Martin. It seems like the amount of multi-threading and the intensity in the chat is unfamiliar to the participants. It is possible that the study leaves this impression out of a desire to showcase that the participants are handling multi-threading confusion with good spirits, and that these chat logs excerpts are atypical.

However, if this is not the case, the multi-threading aspects of the conversation might be seen as more of a nuisance and less of an occasion for humour if the participants had to deal with these problems on a regular basis. It would be interesting to see a similar survey conducted with participants who are familiar with intensive chat or instant messaging. How much effort does handling multi-threading issues require? Even if they can be dealt with in good spirits, they still have to be dealt with, and thus take away focus from the main topic of the conversation.

In conclusion, O'Neill and Martin list three ways of manipulating the turn taking system in text chat.

- Implementing access rules, meaning using some sort of mechanism for selecting the next speaker (similar to designating a new speaker, used in oral conversation). This would reduce the fluidity of the chat. As a remedy, it is suggested that the message that is being typed should be visible to all users as it is being typed, instead of when it is completed and sent.

- Preserving serial adjacency, by letting participants select the statement their message will appear after, as in Smith et al.'s (2000) Threaded Chat. The authors point out that this mechanism disrupts the temporal nature of traditional instant messaging – new messages can appear anywhere on the screen.
- Using multiple chat panes. Similar to the work of Chen et al. (2005), this approach entails separating the statements in the conversation into different panes. This could make it more difficult to keep track of the conversation as a whole, since there are several places on the screen where new messages can appear, similar to the issues facing Threaded Chat.

2.4.2 Visualising message production: Chat circles

Viegas et al. (1999) visualize the production of messages with coloured chat circles (Figure 2.2). As a chat participant types a message, the circle representing that participants starts to grow. Thus, the other participants are aware that a turn in progress. When the message is completed by the sender, it is posted to the screen in a circle, and gradually fades away. Thus, Chat Circles mimics parts of a oral conversation, which has no log transcript saving the conversation history and is focused on what is currently being said.

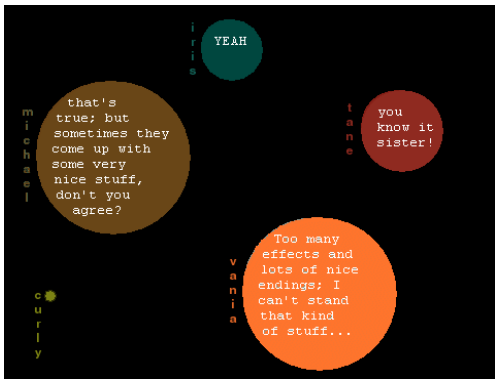


Figure 2.2: Chat Circles screenshot (Viegas et al. 1999).

Chat Circles handles the issue of multiple threads by spatially separating them. The design is tailored to chats with many participants, so that people (represented by circles) cluster and discuss some topic. Only the messages appearing in circles close to you are readable. The other conversations going on are also visible, in that you see the circles expand and fade, but to see the contents of the messages, you have to move away from your current conversation and move adjacent to the other.

This approach would not solve issues of threading within topics, since the spatial arrangement of participants only divides them into topics and does not prevent threading,

2.4.3 Increasing awareness and creating flows: Status Chat and Flow Chat

Vronay et al. (1999) attempt to reduce the problems associated with multi-threading in chat by removing the conditions that caused overlapping threads to begin with. In Status Chat, they provide more information to the conversation participants about what the other participants in the conversation are doing. Primarily, this involves making message composition visible.

Status information is conveyed along with a list of users in a separate pane in the GUI, and next to the user's nickname, the text (s)he is currently composing is visible (Figure 2.3). When the message is completed and sent, it is moved into the main pane and gradually fades from the status pane.

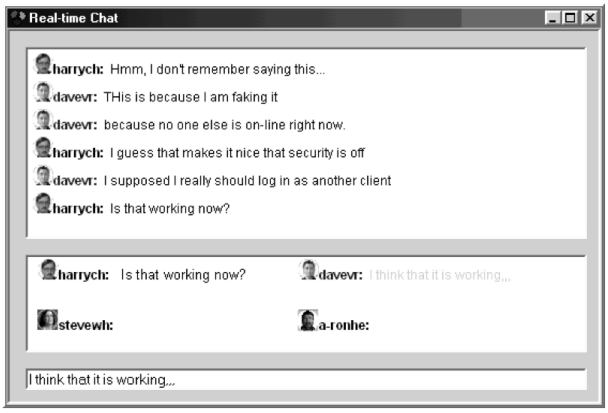


Figure 2.3: Status Chat (Vronay et al. 1999).

Status Chat was compared to an equivalent chat application without the status pane in an experiment with 10 participants divided in two groups. Each group would conduct a conversation using first one application, then the other. The subject of the conversations were either deciding on which 5 videos to bring to a weekend retreat, or which 5 albums to bring along on a road trip.

After the conversations, participants completed a survey with factual questions about the other participant's music and video preferences as well as questions on their subjective experience of the conversation.

Vronay et al. reported that the Status Chat client saw fewer unrelated adjacent statements, because users would see a question as it was being composed and start to type their answer, but delay actually posting it until the question was “committed” to the main chat pane. Thus, the adjacency of relevant statements was preserved, and this led to a subjectively more readable log. It was expected that this would lead to better accuracy on the factual questions in the survey after using Status Chat, but no difference in accuracy was observed.

The authors compared the number of correctional turns in Status Chat to the standard chat, but found no difference – the number of correctional turns was very small. It is worth noting that only 3% of the turns in the Status Chat experiment were classified as “off-topic” by Vronay et al., which is characterized as a very low number.

User acceptance of Status Chat was high; 9 of 10 participants in the experiment preferred Status Chat to the standard chat client.

Vronay et al. observed that temporal information, for instance that a participant started typing earlier than another user, yet sent his message later, was lost in Status Chat. Additionally, Status Chat users had commented that it was difficult to match messages to users in the chat history in that application. This inspired Vronay et al. to create Flow Chat, an application in which the conversation is visualized as a number of tracks (Figure 2.4). Each user has a track with a message composition box to its right. As the message is composed, it is visible in the

message composition box. A message is moved onto the track when it is completed and gradually moves to the left as time progresses. The box surrounding the message illustrates the time span in which the message was composed.

To make the Flow Chat experiment more like a typical chat room conversation, the participants were instructed to try and get to know each other, and were given some suggestions for topics, but no specific instructions. In addition, each conversation had two extra participants who were tasked with keeping the conversation flowing.

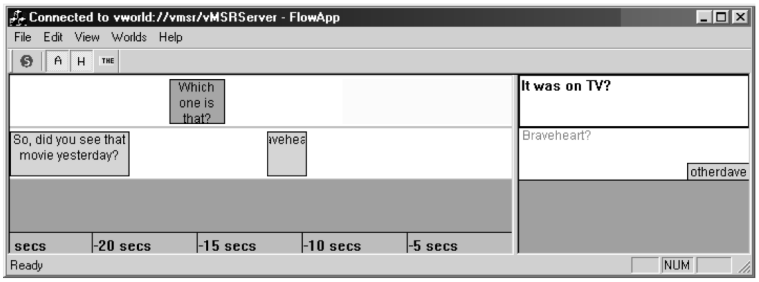


Figure 2.4: Flow Chat (Vronay et al. 1999).

Flow Chat was compared to the standard chat application in four sessions with five participants in each. Vronay et al. found that the chat history function was used more in Flow Chat, and that the share of turns that repeated previously entered information dropped from 5% to 1%.

However, participants disliked the chat history in Flow Chat. Specifically, users felt the text left the screen too quickly, the multiple tracks meant that they had to watch too many different locations on the screen, and the tracks' constant slow movement meant that much of the screen was changing all of the time. Also, users felt uncomfortable with a horizontally instead of vertically oriented chat interface.

Vronay et al. conclude that there are graphical user interface obstacles to overcome, but that viewing chat as a streaming media type has potential. It is also noted that novel concepts such as those introduced in Flow Chat might have to be introduced to users over a longer period of time in order to get valid statistical results when comparing them to traditional chat interfaces.

2.4.4 Creating explicit threads: Threaded Chat

Smith et al. (2000) attempted to improve on standard IM clients with a program supporting threaded conversations. A study with 70 participants who had completed high school but had otherwise varied backgrounds was conducted. The participants were grouped in threes or fours. The groups then assessed three candidates for a position in a fictional firm. Each group participant was given a different piece of information, so that making a correct assessment depended on exchanging information through the chat.

Participants attempted to solve the task in a standard chat client, in an experimental chat client called “Threaded Chat” (Figure 2.5) and a second experimental client which deals with scripted interactions and is not discussed in the paper. “Threaded Chat” structures the chat in a tree, with every conversation thread having its own branch. Replies are thus explicitly connected to the statement they reply to.

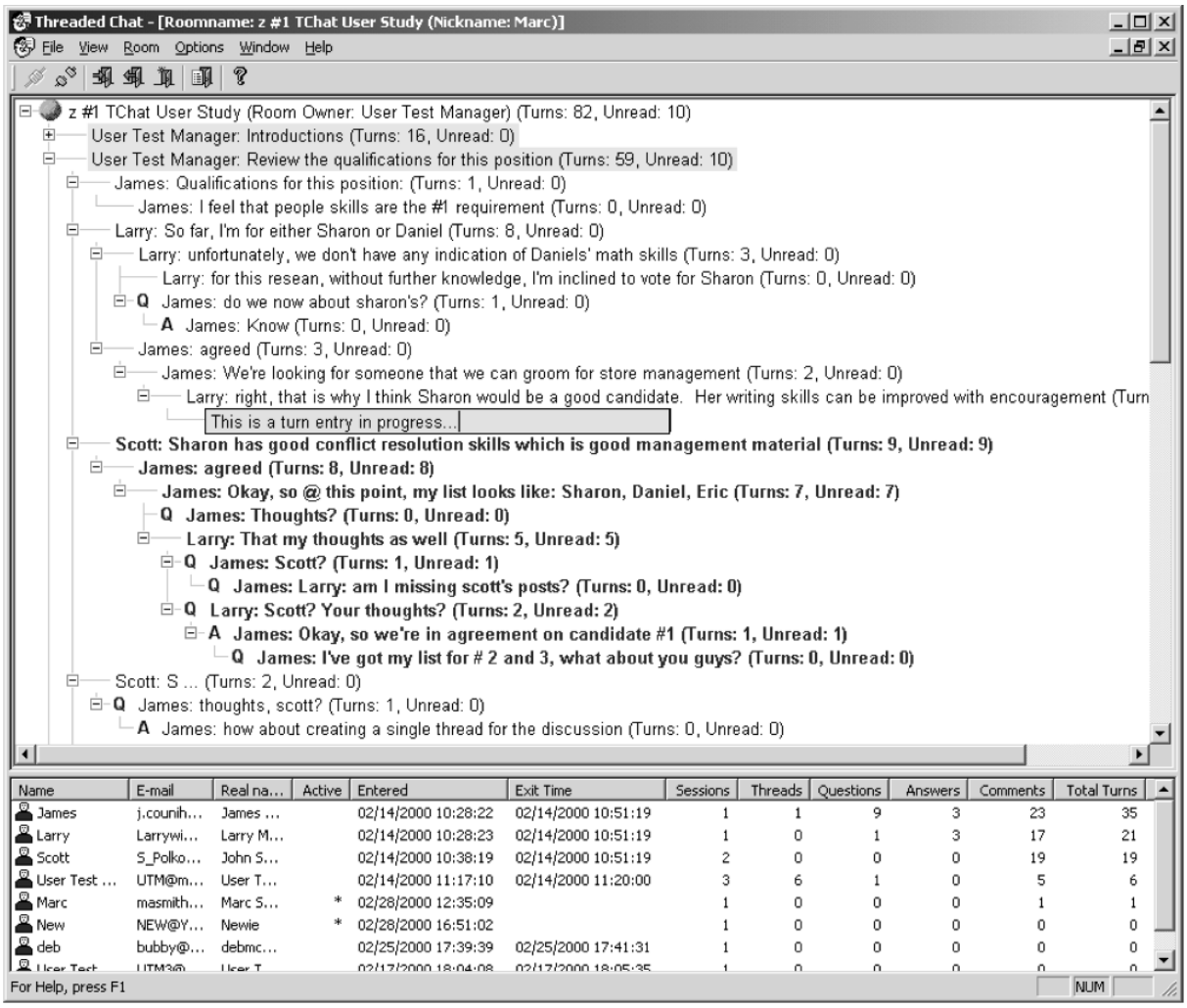


Figure 2.5: Threaded Chat screenshot (Smith et al. 2000).

All the participants attempted the task in all three clients; the order in which they were used was different for different participants to avoid it affecting the outcome of the study. Also, participants were given different roles in each attempt. The discussion was supposed to result in the candidates being ranked in a certain order, so the authors were able to see how effective participants were at coming to the correct conclusion by using the different chat programs.

The authors describe six main issues with instant messaging: lack of links between people and what they say, no visibility of listening in progress, lack of visibility of turns in progress, lack of control over turn positioning, lack of social context, and unstructured log data. We will describe these briefly.

Commonly, IM clients identify people by their nickname, and this precedes their statements in the chat. When messages are sent and received rapidly, it can be difficult to quickly ascertain who said what by reading the chat pane. Some clients attempt to alleviate this by using different fonts or colours for different chat participants.

A user who is typing has no indication as to whether or not the other party or parties in the conversation are actually paying attention to the chat, and what their reactions are. This reduces the sense of social presence in instant messaging, and makes it impossible to adjust the statement according to unseen reactions at the receiving end.

Turns in progress are not visible; most chat programs transmit information only when the user presses the Enter key. This means that chats are not really synchronous; turns appear instantly, in contrast with spoken conversation, where participants can prepare their statements while listening to what the other party is saying. Variations in timing, which people are very attuned to in spoken conversation, can occur in instant messaging for other reasons and can be misinterpreted.

Lack of control over turn positioning means that the only factor determining the placement of the turn is the timing of the "Enter" press. Consequently, after a statement is sent from a user, the other users are in a contest to be the first to address the statement, leading to turns being shorter than what might otherwise have been the case. The short turns can lead to incoherence, and when multiple turns are entered simultaneously and replied to quickly with short turns, users cannot assume that the turn immediately preceding the last one is relevant to it at all. Instead, they must search the preceding lines until finding the one that was actually replied to.

Traditional IM conversations exist only between two or more participants, and though logs may be stored at the different computers, there is no persistence between sessions[8]. The log data might be comprehensible to the chat participants at the time of the chat, but viewed by other people at a later stage, piecing together the conversation from the jumble of lines can be difficult. This also means that a participant in a chat who has been unattentive for a period of time must spend significant time getting back into the chat context.

Threaded Chat attempts to address the problems of poor history logs, lacking social context and turn taking issues by being a combination of chat client and discussion forum. Conversations persist between chat sessions, and they are threaded, so that participants' replies to a chat line (or post) create a branch. The branch can then also be replied to. Turns are thus explicitly connected to the turn they are meant to reply to. A message can also be moved to a different location in the tree if it is misplaced. To address the issue of persistence, the chat server stores statistics for the participants; the number of comments, questions, answers and at what time the user left last if not present at the moment,

The study showed that participants using the threaded chat application used fewer conversation turns to reach their conclusion, which the authors speculate could be caused by more coherent turns being enabled by the threaded chat structure. The level of participation was also more balanced between participants compared to the standard chat program groups; this was in line with the assumptions of the authors – the threaded structure reduced the importance of typing speed in the chat. Users did not have to rush to have their message positioned after the message they were replying to.

Though participants pointed out the usefulness of the threaded chat design, the discussions were usually only following one conceptual thread ("who should we hire"), and Smith et al. speculate that the task they designed was ill-suited to testing Threaded Chat, which should be more useful in conversations with several separate conversation threads going on at once.

[8] A recent feature of Windows Live Messenger, added after Smith et al. published their paper, is that the last part of the log from the previous chat is shown when a new chat with the same participant is started, adding a degree of persistance provided logs are saved.

2.4.5 Arranging messages by topic: IBM's patent

Chen et al. (2005) of IBM applied for a patent on an instant messaging application in which messages can be classified as belonging to a topic either manually by users or automatically by comparing their contents to keywords. The application maintains a "conversation transcript" for the conversation as a whole, and separate panes where the statements classified as belonging to a certain topic are shown, in addition to being shown in the main chat transcript.

The motivation for the topic classification of messages is primarily to make conversations easier to get into for participants who join late in the chat session and to make log transcripts more useful. For instance, chatter about lunch arrangements can be separated from project discussions at the time of the conversation, relieving readers of the log or late joiners of the task of filtering out the noise from the information they are actually looking for. In other words, the motivation for the topic-oriented instant messaging program overlaps with that of Threaded Chat (Smith et al. 2000).

In Figure 2.6, all of the conversation is visible in the leftmost pane, while specific messages classified as belonging to a topic are also shown in topic panes. Topics can have sub-topics.

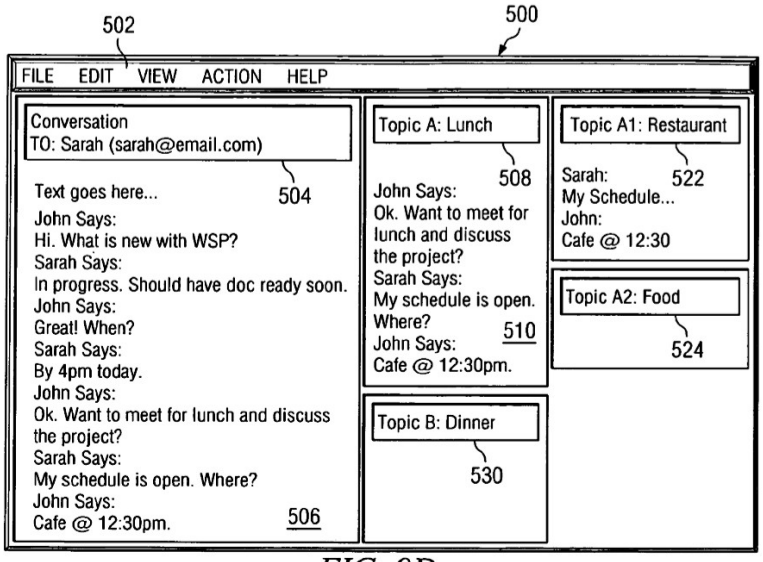


FIG. 9D

Figure 2.6: Chen et al. (2005). One of several variations over the topic separation functionality.

2.5 Summary of solutions to problems of text chat

From the existing research, we extract four main methods for handling the chat conversation problems we have described. The first three are also suggested by O'Neill and Martin (2003).

2.5.1 Explicit threads

Threaded Chat (Smith et al. 2000) allows for separation of different threads in the conversation pane, and makes turn positioning easier. However, the turn positioning control enabled by this system means that the temporal nature of conversation – usually mimicked in chats by the statements being sorted chronologically – is only partly preserved. New statements can appear anywhere in the conversation. Users reported frustation because of this in the Threaded Chat trial, and the perceived quality of discussion was significantly lower in Threaded Chat compared to the standard chat client. Smith et al. discuss possible design choices that could alleviate this problem, such as closing off old branches of the conversation tree automatically and reordering the threads in the user interface. The authors also remark that the client used in the trial had rough edges, and that this probably contributed to the less positive user experience.

It is also worth noting that although being able to select which statement to reply to makes turn-taking easier, it is still possible to compete for the spot directly underneath a statement in a layout similar to that of Threaded Chat – two replies to a message can't share the position immediately beneath the message.

A disadvantage of the Threaded Chat approach is that users' have to perform the additional step of selecting statements whenever necessary. Shen et al. (2006) study algorithms for automatically sorting statements into threads, but this technology is in our view too immature to be a viable solution at this stage.

2.5.2 Different panes for different topics

A topic management system similar to that of Chen et al. (2005) should reduce the intermingling of different topics and consequently reduce the amount of multi-threading. The downside is that activity can happen on several different spots on the screen; however, if the user is primarily interested in only one topic in the conversation, this means that (s)he can ignore the others, while still having the opportunity to look in on them as desired.

Considering the fluid nature of conversation, it seems likely that several threads of conversation will occur inside the scope of one topic, so the problem of thread interruptions will not be eliminated in a topic-oriented chat.

A design with dedicated topic panes could also suffer the disadvantage of increased overhead, since users would have to manually select which topic their statement belongs to, unless a keyword system, such as that suggested by Chen et al. (2005) was used.

2.5.3 Access rules

O'Neill and Martin (2003) suggest rules for turn control as an alternative. For instance, the first user to start typing would have the right to post first. O'Neill and Martin suggest that this specific approach is combined with transmitting every character as it is typed, to keep the chat fluid. Such a mechanism would mimic natural speech, where the first one to “get going” with his/her statement typically is the one who gains the floor and completes his/her statement.

Vronay et al. (1999) implement a “loose” version of this scheme in their Status Chat, where the message composition is visible to all users. Participants are aware of each others' messages as they are produced and can assist each other in turn placement.

Different access rules could also be implemented, for instance, a chat leader could select the next speaker. This would, however, differentiate the system from natural conversation, which is defined as a cooperative activity between participants of equal power.

2.5.4 Radical interface changes

In Chat Circles (Viegas et al. 1999) users effectively “self-moderate”, as users can only follow one topic at the time. Messages are however distributed in a 2D space according to where the participant's representation is situated, so there is no immediately visible connection between turns as in chronological text chat. A separate history view is available.

Flow Chat (Vronay et al. 1999) is another radical chat concept, where the emphasis is on preserving temporal information. It is worth noting that both Threaded Chat and Flow Chat had significant negative reactions from users because of their novel approaches to chat conversation.

3 Designing an improved chat application

3.1 Design goals

There were two main design goals in the work with the prototype client.

First, by considering different instant messaging user interfaces as we developed the prototype, we wanted to try and find a better user interface for text chat conversations. This meant coding some alternative implementations while just discussing and drawing others. Specifically, the chat interface should reduce the amount of multi-threading in the conversations and/or help the user cope with the problems caused by multi-threading.

Second, the final prototype should be advanced enough to serve as a tool to evaluate the new chat interface in an experiment. Thus, it had to be stable enough to handle real usage, have logging functionality to enable log analysis, and be user friendly.

To test the new user interface concept, and not just the implementation, it was essential that the prototype was easy to use. Therefore, the interface was kept simple and as similar to existing chat interfaces as possible. The omission of features not strictly relevant to the chat interface, such as buddy lists and general presence awareness, kept the complexity low and helped create a more controlled environment for the experiment.

3.2 Design process

We considered using a commercial and extendable client, such as Trillian Pro[9], or an open source client, such as the previously mentioned Pidgin, as a starting point for our prototype. In the end, we decided to create a new client. There were several reasons for this,

- The functionality needed to test different chat features did not require IM functionality such as contact lists and presence awareness
- The time required to implement a limited client/server solution would, we estimated, not exceed the time required to learn an existing codebase and modify it
- By writing our own software, we had more flexibility

After deciding to write the program from scratch, we had to decide on a programming platform. We needed a platform with decent GUI capabilities, and it would be an advantageous if we did not have to learn a new language from scratch. Additionally, we would like the program to run on the majority of PC platforms, meaning that the Macintosh, Linux and Windows platforms would have to be supported. As Java is a multi-platform programming language and we had experience in Java programming and some basic knowledge of the Swing GUI framework, we decided to use it as our platform.

We had little knowledge of network programming in Java; for the basic client/server architecture we drew inspiration from Deitel and Deitel (2001), which has a fairly extensive sockets-based client/server example. After creating a basic server, capable of supporting several clients and enabling command-line conversations similar to Unix *talk*, we could start working on the graphical user interface.

[9] Trillian: http://www.ceruleanstudios.com/ (Retrieved April 25, 2007)

A number of different user interface ideas were considered. In the following sections, we introduce the most influential concepts. Throughout the development, we focused primarily on enhancements to the existing typical chat interface. Considering previous research such as Smith et al. (2000) and Vronay et al. (1999), where relatively novel interface concepts were introduced, we believed this would increase the likelihood of emerging with a design that could be compared fairly to the standard chat interface.

3.2.1 StandardMessenger

The first GUI implementation was StandardMessenger (Figure 3.1), which mimics the user interface used in most chat clients.

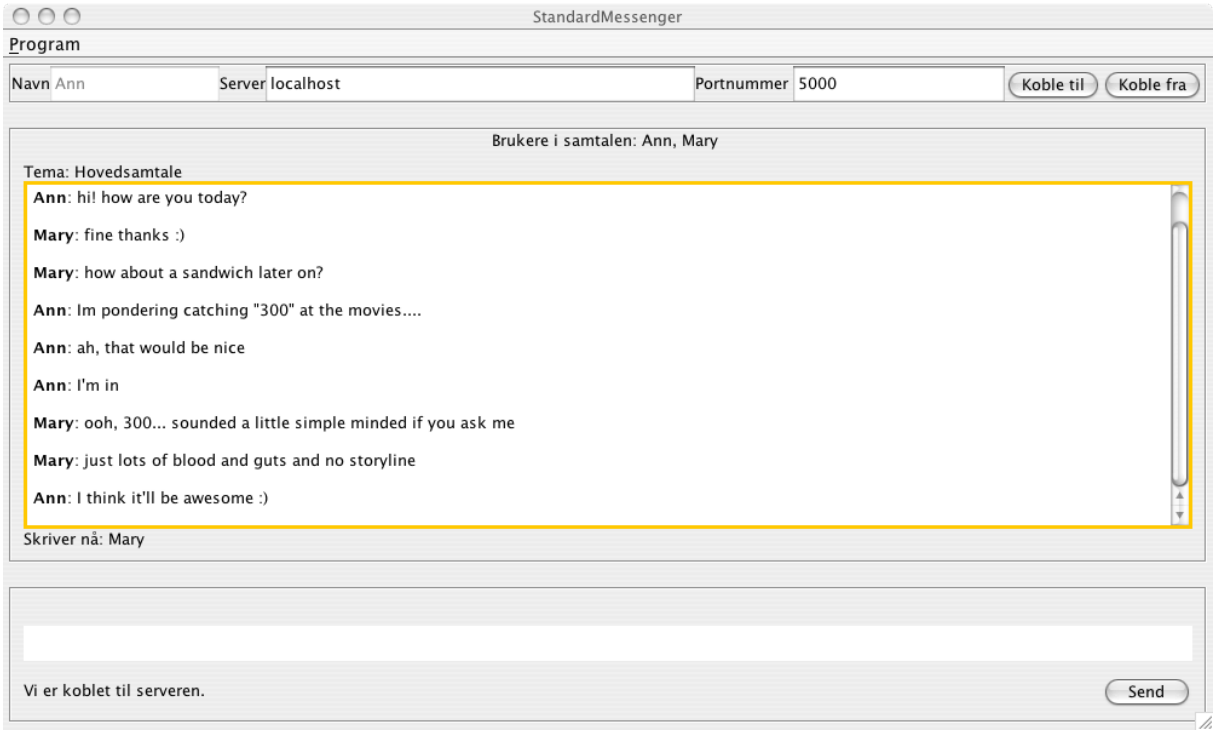


Figure 3.1: StandardMessenger user interface.

Messages are ordered sequentially from top to bottom in the chat pane. Messages are composed in the blank pane at the bottom. Below the chat pane, there is a list of the participants in the conversation who are currently composing a message. Above the chat pane, there is a list of the participants in the conversation.

The features in StandardMessenger are what we see as a minimum for a “modern” chat client, roughly equivalent to the key chat functionality offered by Microsoft's Windows Live Messenger and the Yahoo! Messenger: Sequential listing of the messages in the conversation, you can see who are taking part in the conversation, and you are notified when someone else is typing.

At the very top, there is a pane containing fields for entering nickname and connection information as well as buttons used to connect and disconnect.

This prototype was used as the control client in the experiment.

3.2.2 StandardMessenger with threading

Smith et al.'s (2000) Threaded Chat, where the conversation is structured in a tree, (see screenshot on p. 18) was a natural inspiration for a modified StandardMessenger. We created a version of StandardMessenger with threading support (Figure 3.2), adressing some of the shortcomings Smith et al. mention, namely by highlighting new messages when they entered the tree.

It could have been interesting to compare Smith et al.'s results, where the participants in the experiment were given quite formal tasks, which they executed in sequence, with results from an experiment with more informal conversation and more multi-threading.

However, it was apparent that solving the problem of diverging threads discussed by Smith et al. would require a more complicated GUI. Threads will grow apart as new threads appear at the bottom of the chat while activity continues in threads earlier in the conversation. This means that users will have to scroll up and down in the conversation to watch multiple threads. Additional mechanisms, perhaps involving letting the user spatially reorganize the chat so that threads can be placed next to each other and run in parallel, might be required to solve this problem.

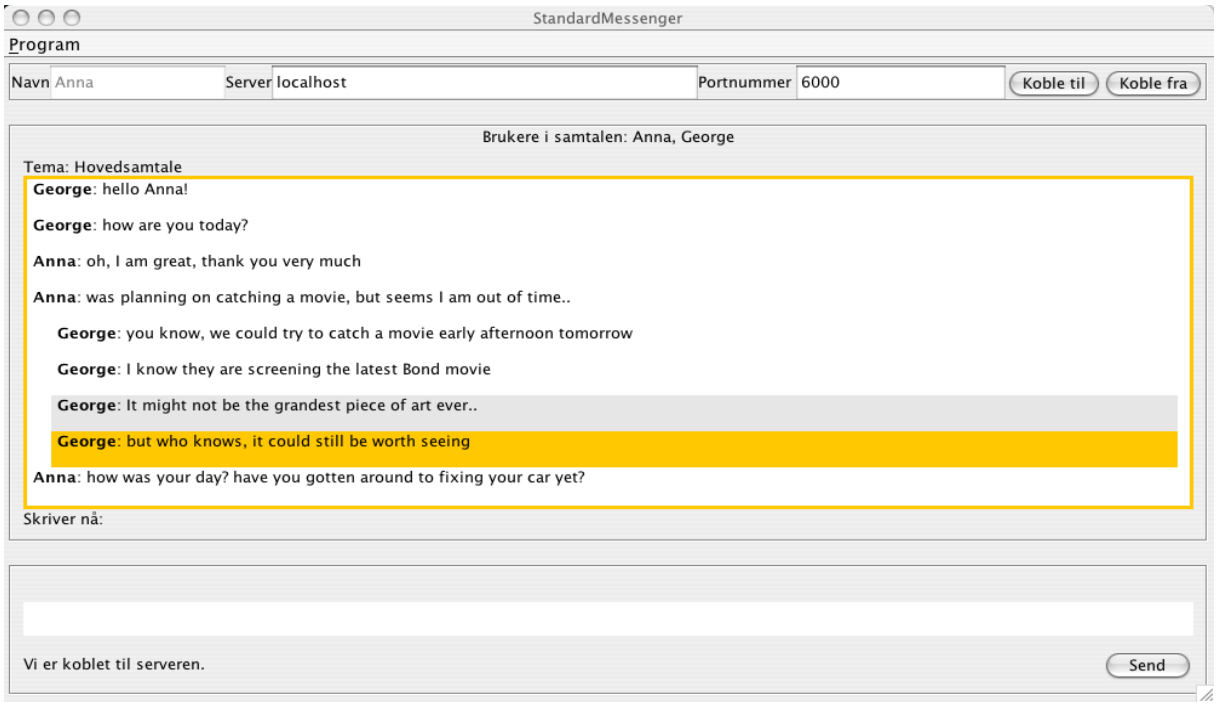


Figure 3.2: StandardMessenger with threading support - not used in the experiment.

3.2.3 OS X Finder-inspired column chat

The file system navigator in Apple's OS X operating system, the Finder, has a feature for navigating folders within folders that was used as an inspiration for a chat GUI.

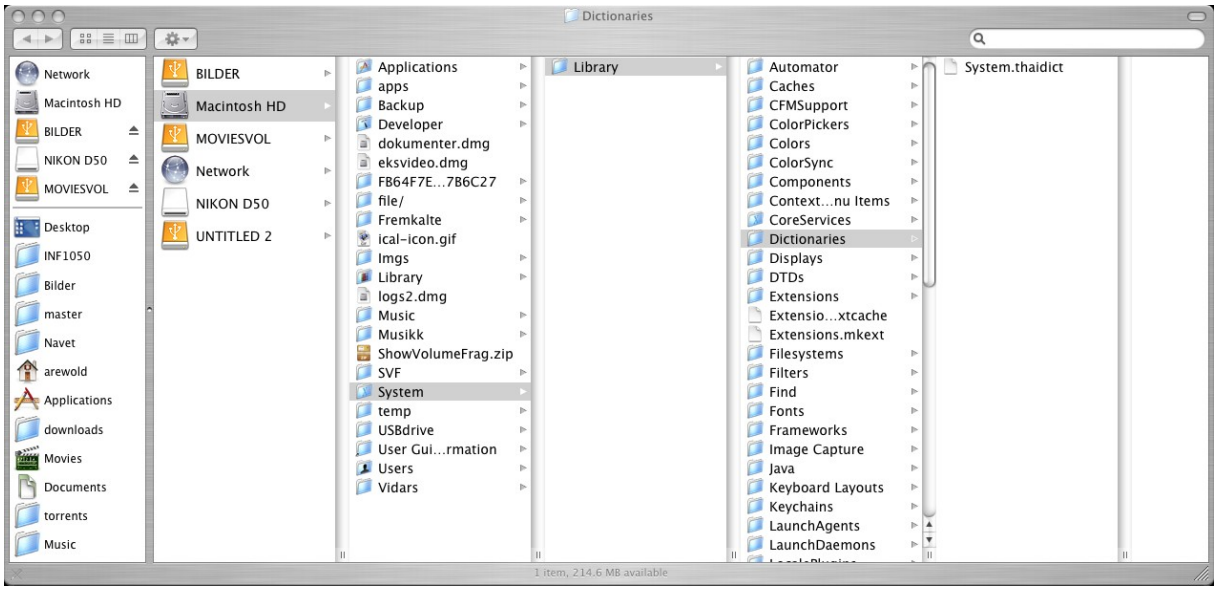


Figure 3.3: The OS X file system navigator, the Finder, in the column view mode.

In the Finder's column view (Figure 3.3), each level in the file system hierarchy is associated with a column, starting with disks and network shares at the left and moving deeper into the directory hierarchy as the user moves to the right. In this screenshot, the user has moved into Macintosh HD/System/Library/Dictionaries, where there is just one file, System.thaidict. If there were more directories in this particular path, the columns on the left would scroll out of sight as new directories are viewed on the right.

The idea was to use the column on the left as a starting point, with messages ordered sequentially from top to bottom in the classic chat fashion. If a user said something that might warrant its own column, any user could click on that message and move it to the column to the right. The conversation would then continue in both columns. This column idea influenced the final TopicMessenger client, which also features a column-oriented design.

3.2.4 TopicMessenger

IBM's topic chat patent (see screenshot on p. 20) and the Finder interface (section 3.2.3) became major inspirations for the TopicMessenger design. The design detailed in the patent from IBM and Chen et al. (2005) has mechanisms for topic administration, a system for controlling what users are allowed to do in terms of renaming and creating topics and automatically sorting statements into topics by keywords.

For TopicMessenger, just the main concept of creating topics and deciding which topic to send a message to was kept. The automatic classification of statements was left out, as we believe this feature warrants a research study of its own. Another difference between the programs is that a statement cannot belong to several topics in TopicMessenger.

Even though the TopicMessenger design handles just topics, and not individual threads, we believed it should significantly reduce the amount of multi-threading.

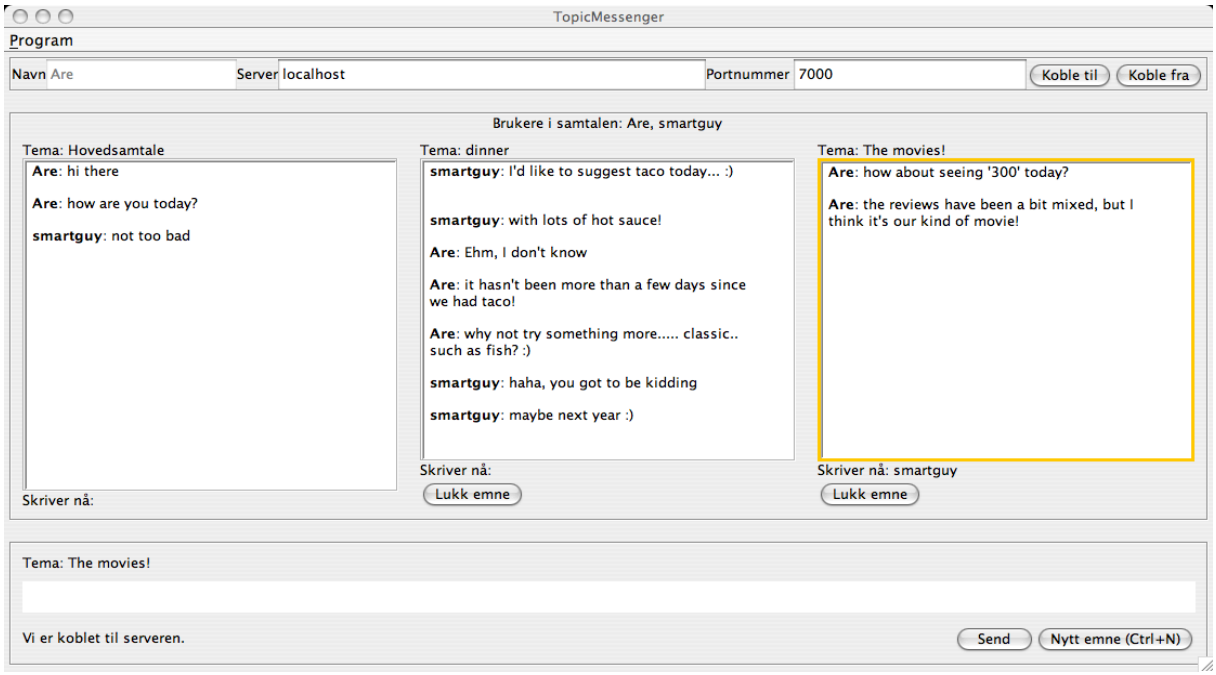


Figure 3.4: TopicMessenger - final version as used in the experiment.

The key feature of TopicMessenger is that the chat pane can be split into several smaller chat panes, where each pane has a topic. When the conversation begins, just one pane, the "Main conversation" pane, is open. This topic pane cannot be closed.

Users can create a new topic pane by clicking the "New topic" button or pressing CTRL+N. The created topic can be closed by clicking "Close topic" below the topic window. The user selects which topic s(he) wants the messages to go into by clicking on that topic pane. The selected topic pane is highlighted with a yellow border. It is possible to move between topic panes using the CTRL key and the arrow keys. The line showing that another user is typing is duplicated across all topic panes, so that it is possible to see which user is active in what topic.

The topic concept is somewhat similar to the channel concept in Internet Relay Chat, where a user can join several channels and, if (s)he wants to, arrange them in a similar fashion to the arrangement in the TopicMessenger screenshot in Figure 3.4. In this screenshot, two additional topic panes have been created. Channels. however, have separate user lists, while all participants in a TopicMessenger conversation automatically take part in all topics in the conversation. When one user creates a new topic, the new area representing it appears at every participant in the conversation. When a user closes the topic, using the button at the bottom of the topic's pane, it is closed only in that user's client, and not in the clients of the other users.

We also wanted to exploit the horizontal screen real estate, which is usually abundant in chat and IM programs. Conversations are often composed of short statements (Smith et al. 2000), requiring at least one line each. IM windows are rarely full-screen by default, so we wanted to use that real estate and provide one pane for each topic.

In the TopicMessenger prototype, users can create as many topics as they like. However, more than 4 or 5 topics is not practical, as the topics become very narrow, even when the program is stretched to occupy the whole screen.

A multitude of topics could also be handled by maintaining their width, but letting the main chat pane increase in size, so that horizontal scrolling would be required to see all of the topics. However, we believe that such a large amount of topics would be too many to keep track of and participate in, so this functionality was left out of the prototype.

TopicMessenger is available on the attached CD in Appendix G.

3.2.5 Combining topics and threads

Combining the thread handling inspired by Threaded Chat and the topic handling from TopicMessenger could potentially eliminate all multi-threading, while alleviating some of the issues associated with Threaded Chat. A version of TopicMessenger was coded where a user could also create threads within each topic by selecting which message (s)he wanted to reply to.

In Figure 3.5, Mary entered her three messages in the left topic before the others started replying. Ann used the threading functionality to reply to Mary's second message. The grey message is the one Ann has chosen to reply to, the orange one just arrived from Mary.

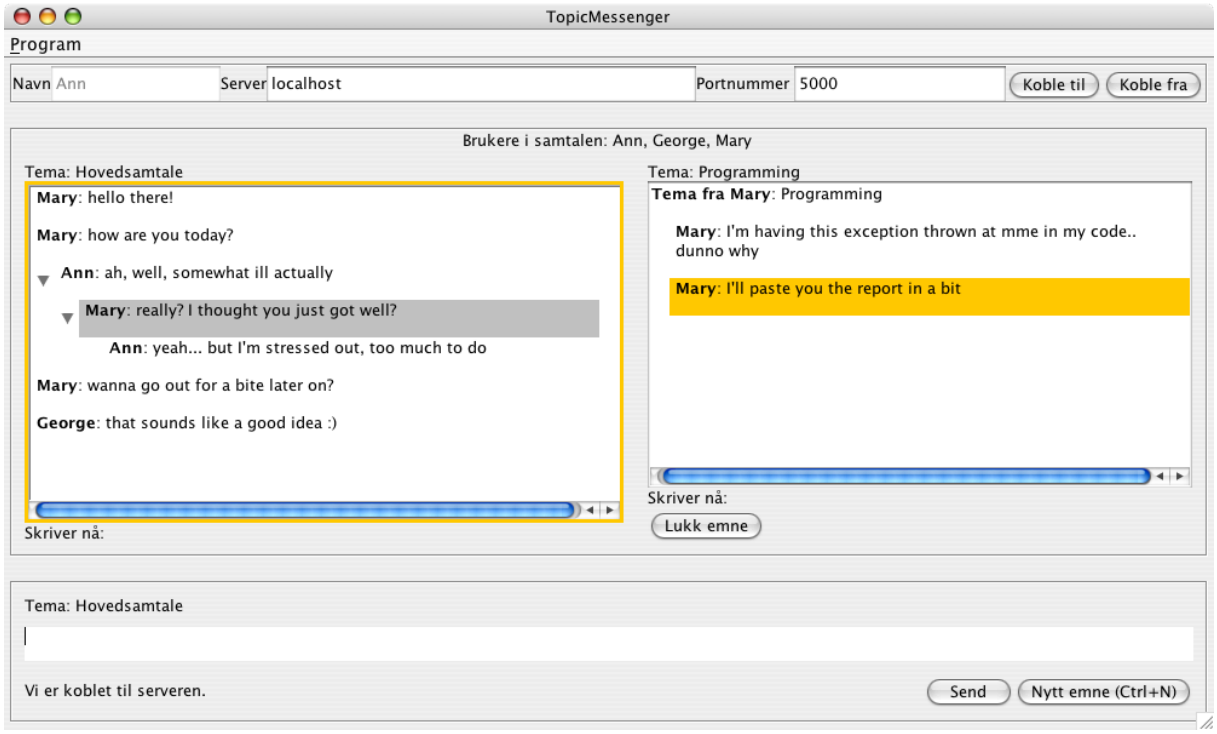


Figure 3.5: TopicMessenger with threading support.

After an informal trial with three participants, we decided to use the pure TopicMessenger design described in the previous section, instead of one with threading. The primary reasons were that the introduction of two new chat mechanisms at the same time might make it difficult for users to take advantage of them, it would be tricky to separate the effects of the two different new mechanisms in the experiment data and the thread organisation challenges discussed in 3.2.2 would probably be too time-consuming to overcome.

Thus, instead of pursuing this path and attempting to test the effect of introducing both topic and thread functionality, we decided to focus on conducting a proper trial of the topic

functionality. This would also partially test the idea of letting threads run in parallel, although in a much more primitive fashion – users would have to define a new topic and then discuss that topic in the assigned topic pane.

3.3 TopicMessenger example scenario

In this section, we show how TopicMessenger was meant to be used by comparing how a fictional conversation can be conducted in StandardMessenger and TopicMessenger respectively.

In StandardMessenger, a conversation between two experienced IM users with decent typing skills and no aversion to multi-threading might look like this:

Ann: Hi goorge!
George: hi ann!
Ann: How are you today?
Ann: wanna head out for dinner later?
Ann: i feel soooo tired
George: I'm not feeling too well either... maybe something's going around
George: dinner would be great
Ann: yeah, my mum was sick last week.. maybe I got it from her
George: how about meatballs in the canteen?
George: yah, that sounds probable
George: remember to wash your hands ;)
Ann: meatballs would be great
Ann: about about 16:30?
George: that's a deal!
Ann: see you then
George: see you :)

In TopicMessenger, with users utilizing the topic functionality, it should look like this:

Topic: Main	**Topic: Dinner**
Ann: Hi george! George: hi ann! Ann: How are you today? [***creates new topic***] Ann: i feel soooo tired George: I'm not feeling too well either... maybe something's going around Ann: yeah, my mum was sick last week.. maybe I got it from her George: yah, that sounds probable George: remember to wash your hands ;)	Ann: wanna head out for dinner later? George: dinner would be good George: how about meatballs in the canteen? Ann: meatballs would be gret Ann: about 16:30? George: that's a deal! Ann: see you then George: see you :)

As the conversation progresses, Ann and George switches back and forth between the topic panes as needed, placing their messages in the correct topic. This should make it more likely that any two statements will be related to each other, and improves conversation coherence.

3.4 Limitations of TopicMessenger

3.4.1 Chat functionality only

The prototypes do not have any sort of buddy list – they are chat clients only, with just a list of users participating in the chat situated above the topic panes. The TopicMessenger client connects to a chat server which only supports one conversation at a time, and all clients connected to the server are placed in that conversation.

3.4.2 Messages cannot be moved

Like in standard chat clients, a message sent to a chat pane cannot be moved around afterwards. Even if the conversation has multiple topics, it is not possible to move a message from one topic to another. Since it is possible to write messages in the wrong topic, functionality for moving messages between topics could be useful.

Such functionality could for instance be implemented using a drag and drop mechanism, where users would simply drag the misplaced message onto the right topic pane. Due to time and resource constraints, no such functionality was implemented in TopicMessenger.

Ideally, functionality that involves moving messages around should be accompagnied by animations showing the messages flowing from a topic to another. This would make it a lot easier to keep track of what goes on in the conversation – in our prototype, changes in the conversation tree (in the threaded version of StandardMessenger) messages just pops in. Ideally, the state changes in the conversation history should be smooth, to make them easy to see and understand for the user.

3.4.3 Requires a lot of screen real estate

As previously noted, TopicMessenger requires an amount of real-estate proportionate with the number of topics in the conversation. It is possible to resize the entire program, but making it too small would make it impractial to have more than one topic.

Consequently, TopicMessenger might require switching back and forth between it and other applications if information visible in other windows is needed when composing messages. “Smaller” clients might allow the user to have the conversation floating in the foreground with so much free space around it that the background content would not be obscured.

The screen space requirement would make it difficult to implement TopicMessenger effectively on small-screen devices like mobile phones. A feature such as tabbing, where only one conversation is visible at a time and the user have to select which of the conversations (s)he wants to see, could make the design workable on a very small screen. It would also be possible to keep the GUI structured as it is, but zoom out, so that users would have to zoom in to view any one topic. The web browser Opera[10] uses this method to help users navigate the world wide web on low-resolution devices such as the Nintendo Wii[11] and mobile phones.

In addition to zooming and tabbing, optical effects such as fisheye techniques (Furnas 2006) could be used to fit the content into a smaller frame while keeping some context information.

---

[10] Opera: http://www.opera.com/ (Retrieved April 15, 2007)

[11] Opera on Nintendo Wii: http://www.opera.com/products/devices/nintendo/wii/screenshots/ (Retrieved April 15, 2007)

3.4.4 Lack of user control over topics

TopicMessenger does not let the user ignore requests for new topics. In other words, when a participant in the conversation opens a new topic, it will appear on the user's screen, whether (s)he wants it to or not.

Also, all topics are of the same size when they are opened, and there is no way to minimize one topic. This is partially by design – the idea is to use the screen space to let the user see several topics simulatenously.

However, in a future prototype, it would be natural to let users ignore new topics if they wanted to, minimize topic panes they were less interested in and also resize individual topic panes.

In the current design, the default or main pane that is present when a conversation starts cannot be closed. The ability to change the name (topic) of the pane or close it would be a natural addition to future versions of the software.

4 Research problem and hypotheses

4.1 Research problem

- Can the quality of leisure instant messaging conversations, as perceived by users, be improved by introducing new methods for structuring the conversation to instant messaging clients?
- Are users willing to take on the added complexity in the user interface caused by such methods, in exchange for improved conversation quality?

4.2 The experiment hypotheses

Through the experiment, we wanted to test two main hypotheses: 1) TopicMessenger conversations will have better conversation quality than StandardMessenger conversations, and 2) Users will express a high degree of acceptance of TopicMessenger.

4.2.1 TopicMessenger conversations will have better conversation quality than StandardMessenger conversations

Supporting or detracting from this hypothesis are a number of sub-hypotheses.

- *The number of interruptions overall will be significantly lower in TopicMessenger compared to StandardMessenger*. We established the concept of an interruption (see definitions on p. 10) as a measure of multi-threading in the conversation, which occurs when several different threads of conversation are intermingled in the chat (Garcia and Jacobs 1998, Smith et al. 2000, Voida et al. 2000, O'Neill and Martin 2003). We believe that the type of multi-threading as seen and discussed in the mentioned papers will be reduced when using our new design.
- *The number of topic interruptions will be significantly lower in TopicMessenger as compared to StandardMessenger*. Given that TopicMessenger has a mechanism for separating statements that belong to a different topic into a different pane, potentially removing all topic interruptions, we expect a significantly reduced number of topic interruptions in TopicMessenger.
- *The number of thread interruptions will be the same in StandardMessenger and TopicMessenger conversations*. Since TopicMessenger does not have a mechanism for handling interruptive statements that are within the same topic as the statement they interrupt, we believe the number of such statements will be about the same in StandardMessenger and TopicMessenger conversations.
- *The number of phantom adjacency pairs will be lower in TopicMessenger conversations compared to StandardMessenger conversations*. Garcia and Jacobs (1998) shows how phantom adjacency pairs occur as a result of multi-threading and statements out of sequence. We believe that the separation of topics allowed by TopicMessenger will reduce the potential for multi-threading in conversations and thus also reduce the number of phantom adjacency pairs. O'Neill and Martin (2003)

observed few occurrences of phantom adjacency pairs, even in a normal chat application. We are interested in seeing whether the results from our experiment will support their observation.

- *Fewer repair statements will be made in TopicMessenger conversations compared to StandardMessenger conversations.* Vronay et al. (1999) and Smith et al. (2000) did not observe any difference between their Threaded Chat and the other chat applications in terms of repair statements; in fact, they observed few repair statements overall. If a significant number of repair statements are made, we believe the number of them in TopicMessenger conversations will be significantly lower than in StandardMessenger conversations, since less multi-threading means less confusion in the conversation and a reduced need for repairs.
- *Users will rate conversations using TopicMessenger as easier to follow compared to conversations in StandardMessenger.* We believe that the confusion that Voida et al. (2002) and Garcia and Jacobs (1999) observes as a consequence of multi-threading in chat conversations will be reduced by our new design.

4.2.2 Users will express a large degree of acceptance of TopicMessenger

In this hypothesis, we drew inspiration from Davis and Venkatesh's paper on pre-prototype user acceptance testing (2004). In the paper, they look at users' opinions on a prototype, expressed in three main terms: Perceived ease of use, perceived usefulness, and behavioural intention.

Perceived usefulness has been shown to be the largest influence on the intention to use a system, and a user's behavioural intention to use a system measured on an early stage in development is predictive of the intention to use it after having hands-on time with a working version of the system (David and Venkatesh 2004).

According to Davis and Venkatesh, perceived ease of use is less important than perceived usefulness in determining the intention to use a system, and it remains unstable until the user has had significant experience with a working version of the system. Considering that ease of use was a design goal for TopicMessenger and that the experiment was intended to test a functional prototype, we included ease of use as a hypothesis.

Although Davis and Venkatesh's study concerns workplace usage, while this experiment has leisure scenarios, and they are focused on pre-prototype testing, while the program used here is a working prototype, we believe the framework they offer is useful for assessing users' opinions towards the software functionality.

Previous research, such as Smith et al. (2000) and Vronay et al. (1999), has tested new interfaces for chat, but not attempted to gauge users' interest in actually making use of the functionality given the opportunity. Through testing the following hypotheses, we want to assess the likelihood that topic functionality would be used by users in a real-world scenario.

- *Participants will perceive the topic functionality as useful.*
- *Participants will express an intention to make use of the topic functionality, if it was available in their instant messaging client.*
- *Participants will perceive the topic functionality as easy to use.*

5 Experiment design

To investigate the research hypothesis, we conducted an experiment. This let us create a controlled environment in which we could compare StandardMessenger and TopicMessenger, record log data and have access to all users during and after the experiment.

5.1 Participants

The experiment had a total of 35 participants. These were drawn from a group of students students, mainly at the University of Oslo, who volunteered to join. Participants were given a gift card worth 100 NOK, and one extra gift card worth 700 NOK was given to one randomly selected participant.

University students were chosen for the experiment because many of them are experienced instant messeenger users and thus in the intended audience for an instant messenger application such as TopicMessenger. We believe that a client such as TopicMessenger is more useful to people who have experience operating instant messenger programs, as it offers conversational features that might be less useful to users who type slowly or who are unfamiliar to how basic instant messaging works.

An additional benefit of using university students is that they are easier to recruit than other potential participants.

The average age of the participants was 23 years, the youngest was 19 and the oldest 30. 25% of the participents were female. On average, participants reported that they had 7.7 years of text chat experience (minimum 3, maximum 16) , and conducted an average of 22 instant messaging conversations every week (minimum 1, maximum 75). The average number of hours spent at a computer per week was 39 (maximum 84, minimum 4).

Although there is a male bias among the participants, which is probably caused by computer science students being over-represented, we believe the participants are a fair representation of young, active instant messaging users, which is the population we believe advanced instant messaging clients will be most useful for.

5.2 Locale

Figure 5.1: The computer lab at the University of Oslo where the experiment took place.

The experiment was conducted in a computer lab at the University of Oslo. Four computers were used – the two on the left side of the room, and the two closest to the camera on the right side (Figure 5.1). Participants were seated so that they could not see the screen of the conversation partner in one on one conversations, and would have to make an effort to see the screen of one of the conversation partners in four-way conversations.

During the experiment, just the participants and the author were present in the room.

5.3 Organization

The experiment was conducted in sessions with four participants. Nine sessions were held in total. In one session, the author had to step in and take part in the experiment (this was caused by a double no-show). The log data from the conversations he took part in are included in the results, while his questionnaire responses are excluded.

The experiment was divided in two main parts. One part tested StandardMessenger, and the other tested TopicMessenger. Whether SM or TM was the first messenger used was random.

Internally in these parts, there were two conversations: One where everyone took part, and one where the participants paired up one on one. The sequence was random here also, so sometimes the group conversation was the first one, and sometimes not.

In the one on one conversations, participants were generally paired with participants they already knew. If none of the participants knew each other, they were assigned partners at random. In both one on one conversations, the conversation participants were the same. The idea was to ease the flow of conversation by assigning people who knew each other to the same conversation, and to compare “apples to apples” by having the same couple of participants converse in both clients in the one on one-conversations.

For each of the four conversations in the experiment, a different set of tasks were given. Two of these tasks were designed for four-way conversations, while two were for two-way conversations. Which test client was used for which task was random.

A time limit of 15 minutes was set on all conversations. The participants were encouraged to take the discussion beyond the scope of their tasks, if they felt like it.

At the start of the experiment, participants were given a demonstration of the test client they were going to use in the subsequent conversations. Equal time (about 2 minutes) was spent introducing both clients.

Before each conversation, the participants were handed a task on a sheet of paper. In two-way conversations, each participant got a different task. In four-way conversations, each task was shared by two participants. The two participants who had been in a two-way conversation with each other were then given the same task.

The objective of the tasks was to give participants something to discuss in the conversations, but also to give them separate agendas. The participants were told that they had 15 minuntes in each conversation. Through different agendas and a time limit, we encouraged multi-threading in the conversation.

After each conversation, a questionnaire was handed out. After the final conversation's questionnaire was completed, a larger questionnaire was handed out. Finally, an informal open-ended group interview was conducted.

5.4 Tasks

As noted, two of the tasks were designed for two-way conversations and two for four-way conversations. A brief summary of the tasks follow.

	Type	Task A	Task B
First prototype	Two-way conversation. Two of these were run in parallel.	Your mobile phone needs replacement. The provided sheet shows some of the models you are considering. Exchange opinions with your conversation partner about which phones are best and try to decide which one you should get.	You'd like to go see a movie tonight, and have an overview of the movie schedule on the provded sheet. You'd like your friend to come with you. Try to agree on a suitable time, movie and location.
	Four-way conversation. Each task shared by two participants.	You and your friend are expecting visitors from abroad. You are planning to show them the best sights of Oslo, and have some suggestions on the provided sheet. As time is short, you have to agree on five suggestions. You very much want to have input from your other two friends on which sights are the most important to see.	You and your friend are planning to arrange a sporting event for foreign students studying at the university. The idea is to combine five sports, preferably typical Norwegian sports. You have some ideas, but want to discuss this with your friends.
Second prototype	Two-way conversation. Two of these were run in parallel.	You and your friend are going to hang out together tonight, and you are planning to watch TV. On the provided sheet you have the TV programme for the evening – try to agree on what to watch from 18 to 23.	You feel like going on a proper holiday abroad soon. You have some travel ideas, which you have written down on the provided sheet, but would very much like your friend's travel advice.
	Four-way conversation. Each task shared by two participants.	You and your friend are both growing tired of the music on your MP3 players, and are very interested in getting some ideas for new music from your two friends.	You and your friend are planning to rent a movie tonight, but you don't really have any firm ideas on which movie you would like to rent. You would like to get some tips and ideas from your two other friends.

5.5 Questionnarie

After each conversation, a form was handed out, asking the participants to rate the client they had just been using on six different statements, using a seven-point scale. The rating was done by circling a number between 1 and 7, where "1" was "I agree" and "7" was "I disagree".

The original questionnaires, in the original language, are included in Appendix D.

- Statements investigating perceived conversation quality after each conversation
  - I was rarely interrupted by the other party in the conversation
  - I never lost overview of the conversation
  - I was always able to make sense of what the other party wrote
- Statements investigating user acceptance after each conversation
  - The functionality in (TopicMessenger/StandardMessenger) contributed to making the conversation efficient
  - If I had access to (TopicMessenger / StandardMessenger) on a day to day basis, I would use it for conversations like this one
  - (TopicMessenger / StandardMessenger) was easy to use

After the final conversation, a larger form was handed out, first asking the participant to compare the two messenger clients on six statements, using a seven-point scale where "1" indicates TopicMessenger and "7" StandardMessenger.

- Statements investigating user acceptance by comparing clients after the experiment
  - In which program were you most interrupted by your conversation partners?
  - In which program was it easiest to maintain a overview of the conversation?
  - In which program was the conversation most coherent?
  - In which program was the conversation most efficient?
  - Which program would you use on a day to day basis?
  - Which program was the easiest to use?

Then, participants rated 10 statements related to acceptance of the topic functionality. Again, they did this by circling a number from 1 to 7, where "1" was "I agree" and "7" "I disagree".

- Statements investigating user acceptance of topic functionality after the experiment
  - I will use the topic functionality if it becomes available in my regular instant messaging program.
  - If functionality for having several topics in the same conversation was available in the instant messaging program I use daily, I would make use of it.
  - Topic support in the instant messaging program I normally use will make me more efficient on a day to day basis
  - Topic functionality will make conversations more productive
  - Conversations will become more efficient with support for several topics in one conversation

- Topic functionality would be useful to have in the messaging program I use on a day to day basis
- The topic functionality in TopicMessenger is clear and understandable
- Using the topic functionality in TopicMessenger does not require a lot of mental effort
- The topic functionality in TopicMessenger is easy to use
- It is easy to understand how the topic functionality in TopicMessenger works

Finally, users were asked some questions about themselves.

- Questions asked to gather information about the participants
  - Age
  - Sex (circle male or female)
  - How many years have you been using instant messaging (ICQ, MSN, Skype, Yahoo! Instant Messaging, Jabber and similar)
  - How many instant messaging conversations do you conduct per week, approximately
  - How much time do you spend instant messaging per week, approximately (hours, minutes)
  - How many hours do you spend in front of the computer per week, approximately

5.6 Questions for the open-ended group interview

- What do you think about the assignments you were given? Did they get you started?
- What did you think about conversing with people in a situation like this?
- What are your thoughts on the topic functionality?
- Would you use this in "real life"?

5.7 Log analysis

All logs from the conversations conducted during the experiment were analysed. Statements were classified into four major categories. Statements such as ":)", "hehe" and "!" were exempt from classification.

5.7.1 Topic interruptions

A statement was classified as a topic interruption if

1. the previous statement is by a different author
2. the statement does not reply to the previous statement
3. the statement addresses a different topic than the previous statement
4. the statement is sent less than 60 seconds since the previous statement

5.7.2 Thread interruptions

A statement was classified as a thread interruption if

1. the previous statement is by a different author
2. the statement does not reply to the previous statement

3. the statement addresses the same broad topic as the previous statement
4. the statement is sent less than 60 seconds since the previous statement

5.7.3 Repair statements

A statement was classified as a repair statement if was only written to clear up an earlier misunderstanding due to statements not being placed adjacent to the statement they were replying to.

5.7.4 Phantom adjacency pairs

A pair of statements were classified as phantom adjacency pairs if they, judging by the conversation log up to the point where they were made, appeared to be related to each other, but further inspection of the log revealed that they were in fact not.

5.8 Statistical analysis

To analyse the data gathered through the questionnaires and the log analysis, we relied primarily on within-subjects one-tailed t-tests (Moore and McCabe 2003) where data related to StandardMessenger and TopicMessenger conversations was compared. For the final questionnaire, descriptive statistics were used.

Our data did not indicate that the underlying population of instant messenger users is normally distributed. Since we have a very small sample size, at only 35 participants, it is possible that our data does not reflect the real population distribution.

A normally distributed population is one of the assumptions of the t-test. The t-test is, however, quite resilient to violations of this assumption, and as we are most familiar with it, it is our primary statistical test in this thesis. To increase the confidence in our findings, we have also conducted Wilcoxon signed-rank tests on our data (Wilcoxon 1945). The Wilcoxon signed-rank test is a non-parametric test and does not assume that the underlying population is normally distributed. It is a more powerful test for heavily skewed populations and recommended when the population characteristics are unknown and the sample size is small (Bridge and Sawilowsky 1998). The results from the Wilcoxon signed-rank tests were similar to the t-tests we ran and are available in Appendix F.

More specifically, the t-tests were used to compare the number of interruptions in conversations conducted with TopicMessenger and StandardMessenger, as well as participants' perceptions after using TopicMessenger and StandardMessenger respectively. These tests were conducted on pairs or sessions. When using pairs, the average of the values from the two participants in a conversation in one messenger was compared to the average of the values from the same participants' conversation in the other messenger. When using sessions, the values from all four participants in a four-way conversation in one messenger were averaged and compared to the averaged values from the same four participants in a conversation with the other messenger.

However, since the two or four participants in a conversation can influence each other, we also conducted separate t-tests using data from individual users who did not participate in the same experiment sessions. This yields a more thorough look at the data we have collected. Generally, these t-tests gave the same results as the aggregate ones, and they are included in Appendix E.

Similarly, for the data resulting from the final questionnaire, we examined the descriptive statistics per session and for the population as a whole. Again, the results were the same.

6 Experiment results

In this chapter, we present the results from the experiment. We begin with results related to the number of interruptions, repair statements and phantom adjacency pairs in the conversations, before moving on to perceived conversation quality and user acceptance of the topic functionality.

6.1 Overall number of interruptions

We hypothesised that the total number of interruptions would be lower in TopicMessenger conversations compared to StandardMessenger conversations. The total number of interruptions is the sum of topic interruptions and thread interruptions. We will compare these sums for two-way and four-way conversations separately.

6.1.1 Two-way conversation

	Mean number of interruptions per minute	Standard deviation	t	p
StandardMessenger	1.527	0.468	-3.6	< 0.001
TopicMessenger	0.865	0.198		

Table 6.1: Comparison of the average number of interruptions per minute in TopicMessenger and StandardMessenger two-way conversations.

Among the 18 conversations in the experiment, the average number of topic interruptions in StandardMessenger was 8.7. The average number of thread interruptions was 14, giving a total average of 22.7.

The average for the 18 conversations conducted using TopicMessenger was 2.05 topic interruptions and 11.6 thread interruptions respectively, for a total average of 13.65.

To take into account the slightly varying length of the conversations (SM average duration = 949 seconds, TM average duration = 909 seconds) we calculate the number of interruptions per minute of conversation for each conversation and compare these two distributions using Student's t-test.

By conducting a one-tailed within-subjects T-test of the data, we get a p-value of 0.001 (t = -3.6), indicating that the difference in interruptions is statistically significant.

Could the increased number of interruptions in StandardMessenger be caused by more intensive conversations? After all, the interface imposed by TopicMessenger could potentially take some time away from typing and thus result in slower conversations. To examine this, we gathered the number of messages, words and characters per minute for each conversation.

The average number of characters per minute was 118 in StandardMessenger conversations (standard deviation 30) and 120 in TopicMessenger conversations (standard deviation 25). The average number of messages per minute was 3,59 (SD 1,28) for StandardMessenger and 3,57 (SD 1,04) for TopicMessenger. The difference is not significant.

Thus, the data indicates that using TopicMessenger results in two-way conversations with fewer interruptions, supporting the hypothesis.

6.1.2 Four-way conversation

	Mean number of interruptions per minute	Standard deviation	t	p
StandardMessenger	4.133	1.579	-7.358	< 0.001
TopicMessenger	1.891	0.387		

Table 6.2: Comparison of the average number of interruptions per minute in TopicMessenger and StandardMessenger four-way conversations.

Among the 9 four-way conversations in the experiment, the average number of topic interruptions in StandardMessenger was 22.2; the average number of thread interruptions was 46. The corresponding numbers for TopicMessenger were 3.33 and 26.

The average total number of interruptions were thus 48.8 for StandardMessenger and 29.33 for TopicMessenger.

To take into account the slightly varying length of the conversations (SM average duration = 965 seconds, TM average duration = 931 seconds) we calculated the number of interruptions per minute of conversation for each conversation and compared these two distributions using Student's t-test.

By conducting a one-tailed within-subjects T-test of the data, we get a p-value < 0.001 (t = -7.358), indicating that the difference in interruptions is statistically significant.

Again, it is possible that the larger number of interruptions could be caused by higher chat intensity. The average number of characters per minute is 84 and 86, respectively, and the standard deviations are 16.6 and 15.6. In other words, the conversation intensity remained about the same across clients.

The hypothesis is supported – the data indicates that the number of interruptions is significantly smaller in conversations using TopicMessenger.

6.2 Number of topic interruptions

We hypothesised that TopicMessenger conversations would have a smaller amount of topic interruptions compared to StandardMessenger conversations. If the experiment participants use the topic functionality of TopicMessenger to separate topics into different chat panes, it would seem reasonable to expect fewer occasions of topic interruptions in the conversation.

To test this hypothesis, we compared the number of topic interruptions per minute for two-way and four-way conversations respectively. Note that, as previously mentioned, the conversation intensity as measured by average characters per minute is about the same for StandardMessenger and TopicMessenger conversations.

6.2.1 Two-way conversations

	Mean number of topic interruptions per minute	Standard deviation	t	p
StandardMessenger	0.596	0.464	-4.096	< 0.001
TopicMessenger	0.129	0.111		

Table 6.3: Comparison of the average number of topic interruptions per minute in two-way StandardMessenger and TopicMessenger conversations.

The average number of topic interruptions per minute in TopicMessenger conversations was relatively low, at 0.129. Four conversations suffered no topic interruptions at all, while one conversation had 7 topic interruptions. The average number of interruptions in the 18 TM conversations was 2.06.

StandardMessenger conversations had an average of 8.78 interruptions and 0.596 per minute. The "record" was 22 interruptions in one conversation, while two conversations had no interruptions at all.

We conducted a within-subjects one-tailed T-test of the topic interruptions per minute data (N=18). The resulting p-value was < 0.001 (t = -4.096), indicating that the increased number of topic interruptions in StandardMessenger is statistically significant, and supporting the hypothesis.

6.2.2 Four-way conversations

	Mean number of topic interruptions per minute	Standard deviation	t	p
StandardMessenger	1.126	0.633	-4.821	< 0.001
TopicMessenger	0.215	0.161		

Table 6.4: Comparison of the average number of topic interruptions per minute in four-way StandardMessenger and TopicMessenger conversations.

The average number of topic interruptions per minute in TopicMessenger four-way conversations was again relatively low, at 0.215. The average number of interruptions per conversation was 3.33. One conversation had no topic interruptions, while the maximum was 8.

StandardMessenger conversations had an average of 20.22 interruptions and 1.126 per minute. The minimum and maximum values were 8 and 35.

We conducted a within-subjects one-tailed T-test of the topic interruptions per minute data (N=9). The resulting P-value was < 0.001 (t = -4.821), indicating that the increased number of topic interruptions in StandardMessenger is statistically significant, and supporting the hypothesis.

6.3 Number of thread interruptions

Since TopicMessenger only provides a mechanism for separating statements into different topics, and not into different threads, we hypothesised that the number of thread interruptions

would be about the same in TopicMessenger conversations as in StandardMessenger conversations. With TopicMessenger, the topic interruptions are avoidable – the statement should just be placed in a different chat pane in the program interface – but the thread interruption problem remains.

To test this hypothesis, we compared the number of thread interruptions per minute for two-way and four-way conversations respectively. We also conducted a t-test, however, as a t-test is used to investigate whether the mean of two distributions differ, we do not rely on it to test this hypothesis, but supply it for its information value.

6.3.1 Two-way conversations

	Mean number of thread interruptions per minute	Standard deviation	t	p
StandardMessenger	0.939	0.560	-1.586	0.13
TopicMessenger	0.737	0.440		

Table 6.5: Comparison of the average number of thread interruptions per minute in two-way StandardMessenger and TopicMessenger conversations.

The average increase in the number of thread interruptions per minute is 27% in StandardMessenger over TopicMessenger.

We conducted a within-sample two-tailed T-test of the thread interruptions per minute data (N=18). The resulting p-value was 0.131 (t = -1.586), indicating that the differing number of thread interruptions in StandardMessenger is not statistically significant. There is a difference between SM and TM conversations, but it is not large enough to be conclusive.

6.3.2 Four-way conversations

	Mean number of thread interruptions per minute	Standard deviation	t	p
StandardMessenger	2.872	1.187	-4.358	0.002
TopicMessenger	1.675	0.555		

Table 6.6: Comparison of the average number of thread interruptions per minute in four-way StandardMessenger and TopicMessenger conversations.

On average, the number of thread interruptions per minute is 72% higher in the StandardMessenger client. The t-test of these data gave a p-value of 0.002 (t = -4.358).

The numbers show a clear difference between the number of thread interruptions in StandardMessenger and TopicMessenger, disproving our hypothesis.

6.3.3 Conversation intensity

At this stage, we decided to analyse data on conversation intensity in StandardMessenger and TopicMessenger conversations, to look for possible causes for the reduction in thread interruptions.

It would seem probable that conversations with four participants are more intensive, as measured in total characters per minute (CPM) *per conversation*, not per participant, than two-way conversations, considering the number of users is doubled. Of course, characters aren't responsible for interruptions – messages are – so we would like to compare the number of messages per minute also. Maybe the intensity increase is larger when going from SM two-way to SM four-way conversations than from TM two-way to TM four-way?

	Mean CPM per conv.	SD	Mean # of messages per minute per conv.	SD	Mean # words per message per user	SD
SM, two-way	240.72	25.73	7.14	1.03	7.08	1.88
TM, two-way	237.68	30.38	7.18	1.25	7.09	1.46
SM, four-way	343.84	15.55	10.51	0.64	6.63	1.96
TM, four-way	337.39	16.59	10.27	0.66	6.49	1.79

Table 6.7: Conversation intensity in two-way and four-way StandardMessenger and TopicMessenger conversations.

The amount of messages in a conversation is not directly proportional with the number of participants, but at 42%, the increase is still very noticeable. The intensity is similar in the different prototype clients, and the percentage increase when going from two to four participants is about the same in both TM and SM, indicating that the choice of messenger is less important for the intensity than the number of people in the conversation.

To investigate the possible connection between intensity and interruptions, we created scatterplots and calculated the correlations for the data for the four different conversation classes – two-way and four-way StandardMessenger and TopicMessenger.

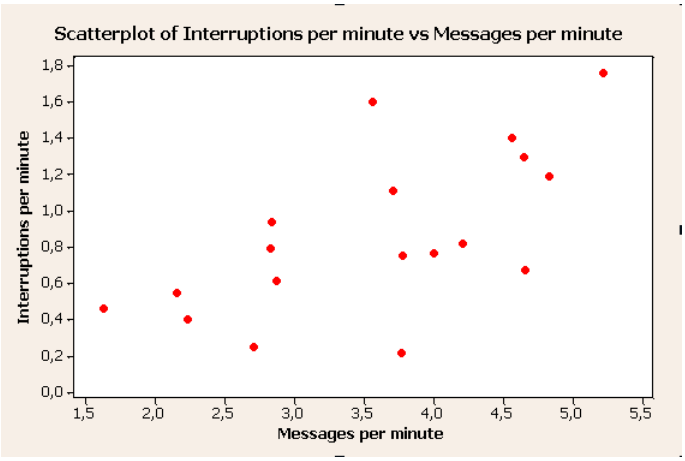


Figure 6.1: Scatterplot of conversation intensity versus the number of interruptions per minute in two-way StandardMessenger conversations.

The scatterplot (Figure 6.1) suggests a relationship between conversation intensity and interruptions. The calculated correlation is 0.64 ($p = 0.004$), indicating the relationship is significant.

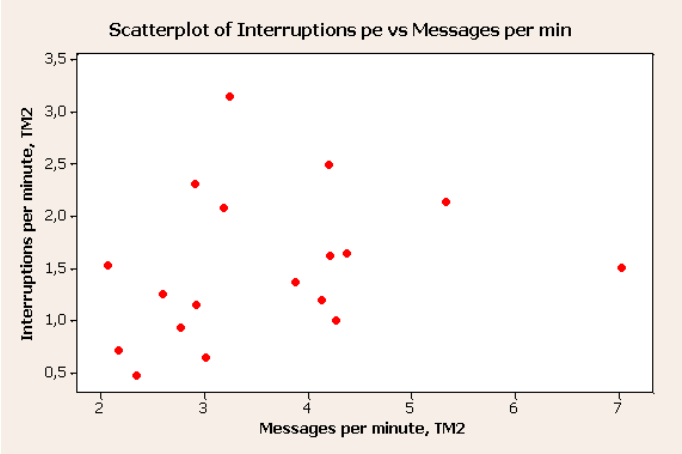


Figure 6.2: Scatterplot of conversation intensity versus the number of interruptions per minute in two-way TopicMessenger conversations.

Looking at the scatterplot for two-way TopicMessenger conversations (Figure 6.2), there does not seem to be a obvious association between messages per minute and the number of interruptions. The correlation is weak, at 0.255 (p= 0.307).

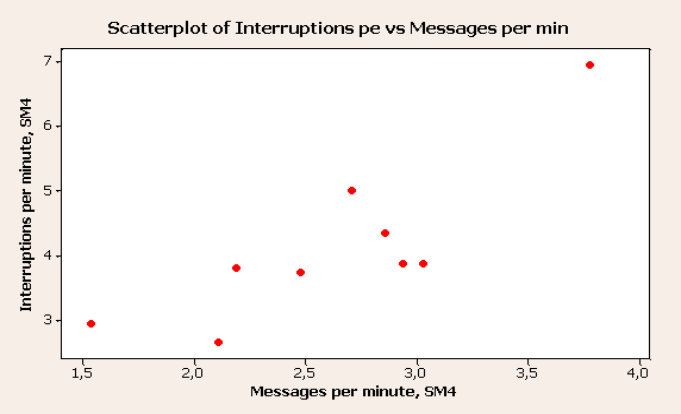


Figure 6.3: Scatterplot of conversation intensity versus interruptions per minute in four-way StandardMessenger conversations.

The data points for four-way StandardMessenger conversations appear to show a linear relationship between the number of messages per minute and the number of interruptions (Figure 6.3). The correlation is 0.839 (p = 0.005). Strengthening this association is the data point from Session 9, with 3.78 messages per minute and 6.94 interruptions per minute.

If we set the upper limit for outliers at the third quartile plus 1.5 times the interquartile range, the limit is 4.24 for messages per minute and 6.685 for interruptions. This classifies the Session 9 data as an outlier. Removing it reduces the correlation to 0.683 (p = 0.06). Considering that this outlier fits within the overall pattern of the data and supports the association already demonstrated in the StandardMessenger two-way data, we believe that it is valid to include it here.

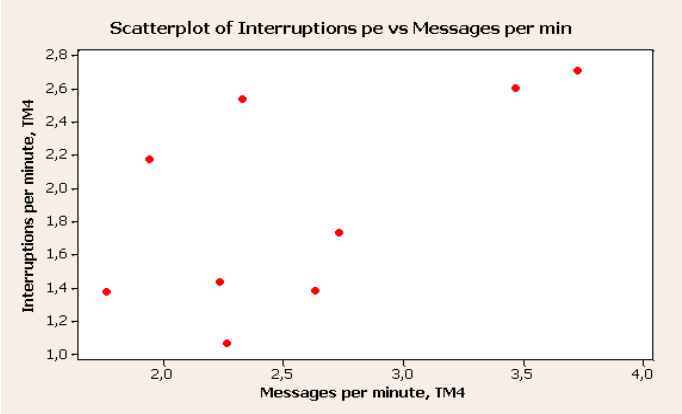


Figure 6.4: Scatterplot of the conversation intensity versus the number of interruptions in four-way TopicMessenger conversations.

From the scatterplot for four-way TopicMessenger conversations (Figure 6.4), the association between messages per minute and the number of interruptions is unclear – in line with what we might expect given the figures for two-way TopicMessenger conversations. The correlation is 0.655 ($p = 0.055$); the small number of data points makes finding a significant correlation difficult. If we had more data, we might be able to classify the data from Session 3 and Session 7 as outliers. The conversations in these sessions had a relatively high number of interruptions per minute, but also a low intensity.

We note the weaker correlation between intensity and interruptions seen in TopicMessenger conversations versus StandardMessenger ones as a possible partial explaination of the reduction in both thread and topic interruptions.

6.4 Number of phantom adjacency pairs

We hypothesised that the number of phantom adjacency pairs would be lower in TopicMessenger conversations as compared to StandardMessenger. The data shows that just two conversations experienced more than one occurrence of phantom adjacency pairs; both of these were four-way conversations using TopicMessenger. The hypothesis is not supported.

6.5 Number of repair statements

We hypothesised that the number of repair statements would be reduced in TopicMessenger conversations compared to StandardMessenger conversations. In the log data from the two-way conversations in the experiment, just two instances of repair statements were found, both in conversations conducted with StandardMessenger.

In the four-way conversations, a total of two repair statements were found in StandardMessenger conversations, versus four in TopicMessenger conversations.

From the few occurrences of repair statements found, we cannot say whether TopicMessenger or StandardMessenger is preferable in this regard. The hypothesis is not supported.

6.6 Conversations will be easier to follow in TopicMessenger

We hypothesised that users would rate conversations using the topic-enabled client as easier to follow compared to conversations in the single-topic client. This hypothesis was tested with questionnaires, one given after each conversation and one at the end of the experiment. We will examine the results from these separately.

Since conversation quality as experienced by one party in the conversation will be affected by the other party in the conversation, we compared both averages gathered from conversations (two-way and four-way respectively) in the different messenger prototypes as well as the scores from individuals who did not take part in the same conversations. Apart from variations that are expected when analysing small numbers of data points, the results from the tests with independent values were similar and are available in Appendix E.

6.6.1 After two-way conversations

	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, pairs
I was rarely interrupted...	3.03 (1.28)	2.36 (0.87)	t = -2.166 p = 0.022
I never lost overview...	2.17 (1.36)	2.33 (0.92)	t = 0.566 p = 0.289
I was always able to make sense of...	2.28 (1.45)	2.08 (0.93)	t = -0.458 p =0.326

Table 6.8: Perceived conversation quality after two-way conversations.

The hypothesis is that users will rate the conversation as easier to follow in the TopicMessenger client. This would to lower numbers for TopicMessenger in the questionnaire.

The data shows that on average, users experienced a higher level of conversation quality in TopicMessenger with two of the three criteria. The only criteria where the difference is significant for conversations is for the statement “I was rarely interrupted by the other party in the conversation”.

For the second criteria, StandardMessenger scores better than TopicMessenger, but the standard deviation for both sets of data are high, and the difference is not significant.

The data is ambiguous as to whether the users perceived TopicMessenger as easier to use versus StandardMessenger. The hypothesis is not supported.

6.6.2 After four-way conversations

	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, session level
I was rarely interrupted...	5.875* (0.25*)	3.167 (0.74)	t = -10.078 p = 0.001*
I never lost overview...	4.583 (0.919)	3.694 (0.982)	t = -1.724 p = 0.061
I was always able to make sense of...	4.083 (0.673)	2.944 (0.527)	t = -3.861 p = 0.002

Table 6.9: Perceived conversation quality after four-way conversations.

* The data for conversations 1 through 5 was lost, so the results for these StandardMessenger conversations is based on just 4 conversations. Consequently, the T-test is based on just the four last group conversations conducted in TopicMessenger and StandardMessenger.

As with the two-way test, the hypothesis is that users will rate the conversation as easier to follow in the TopicMessenger client. This would correspond to lower numbers for TopicMessenger in the questionnaire.

The data shows that, on average, users reported higher conversation quality in all three criteria. At the 5%-level, the conversational-level results are significant for “I was rarely interrupted by the other party in the conversation” and “The other party's statements always remained coherent to me”.

That half the data points are missing for the first criteria reduces the credibility of our t-test. The t-test for the second criteria does not show a significant difference at the 5% level. The difference on the third criteria is significant, with a p-value of 0.002.The hypothesis is partly supported, and the data does indicate that the users perceived conversations in TopicMessenger as more coherent than those conducted in StandardMessenger.

6.6.3 At the end of the experiment

In the following table, “fully favour” indicates that users circled “1” on the scale from 1 to 7, meaning that they were as positive as possible towards TopicMessenger. “More or less favour” means that they circled 1, 2 or 3, and supported TopicMessenger to one degree or another. The latter group thus includes the group who “fully favour” a given statement.

	Mean	SD	% more or less favor TM		% fully favour TM	
			Users	Sessions	Users	Sessions
1. In which program were you most interrupted by your conversation partners?	5.46	1.6	74%	89%	31%	22%
2. In which program was it easiest to maintain a overview of the conversation?	2.74	1.85	69%	66%	37%	0%
3. In which program was the conversation most coherent?	2.49	1.72	83%	89%	43%	11%
4. In which program was the conversation most efficient?	3.34	1.81	57%	78%	14%	0%

Table 6.10: Users' comparison of conversation quality in TopicMessenger and StandardMessenger after the experiment.

On all statements, the majority of participants, both individually and by fours, favoured TopicMessenger, although there were large variations. On the question of "efficiency", only a slight majority, 57% favored TopicMessenger, and just 14% had an undivided preference for TM over StandardMessenger on this question.

However, 83% thought TopicMessenger conversations were more coherent than StandardMessenger ones to some degree and 74% felt they were less interrupted in TopicMessenger compared to StandardMessenger. Grouped into sessions, the support for TM on these questions rise to 89% respectively.

In sum, the results show that users perceived TM conversations as easier to follow, supporting the hypothesis.

6.7 Users will perceive the topic functionality as useful

After each conversation, users were asked whether they thought the functionality in the messenger prototype they had just used contributed to making the conversation efficient. At the end of the experiment, they were asked about topic functionality in general.

We focus primarily on the answers given at the end of the experiment, as these relate to topic functionality in general and are the most relevant for our hypothesis, but include the comparisons between the prototypes to increase our understanding of users' perceptions.

The answers were given on a scale from 1 to 7, where 1 was "Agree" and 7 "Disagree". In the tables assessing user acceptance by share, the percentage for "more or less agree" includes participants who circled 1, 2 or 3, and thus includes the group who circled 1 and completely agrees with the statement.

6.7.1 After two-way conversations

	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, pairs level
The messenger's functionality contributed to making the conversation more efficient	3.69 (1.42)	2.83 (1.01)	t = -2.01 p = 0.03

Table 6.11: Perceived efficiency contributions of functionality in two-way StandardMessenger and TopicMessenger conversations.

Users perceive TopicMessenger's functionality as making a more positive contribution to the conversation efficiency than StandardMessenger's functionality.

6.7.2 After four-way conversations

	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, session level
The messenger's functionality contributed to making the conversation more efficient	5.2 (0.77)	3.09 (0.71)	t = -7.63 p < 0.001

Table 6.12: Perceived efficiency contributions of functionality in four-way StandardMessenger and TopicMessenger conversations.

After four-way conversations, users are less positive to the effects of both clients' functionality on the conversation efficiency compared to the two-way conversations, but StandardMessenger suffers significantly more than TopicMessenger.

6.7.3 At the end of the experiment

These results are also available as charts in Appendix B.

	Mean	SD	% more or less agree		% fully agree	
			Users	Sessions	Users	Sessions
1. Topic support will make me more efficient	3.9	1.3	31%	33%	0%	0%
2. Topic functionality makes conversations more productive	3	1.52	74%	67%	20%	0%
3. Topic functionality makes conversations more efficient	2.89	1.39	77%	100%	14%	0%
4. Topics would be useful to have in everyday use	2.89	1.59	74%	89%	20%	0%

Table 6.13: Perceived topic functionality usefulness at the end of the experiment.

With an average score of 3.9 and a standard deviation of 1.3 the score for the first statement does not support the hypothesis. Just 31% somewhat agreed that topic support would make them more efficient on a day to day basis.

The average score for the statement “Topic functionality will make conversations more productive” was 3, with a standard deviation of 1.52. 74% somewhat agreed to the statement, while 20% agreed fully.

Support for the third statement, “Topic functionality makes conversations more efficient”, was relatively strong, with an average score of 2.89. 77% agreed to some extent, while 14% agreed fully.

In assessing the usefulness of the functionality, the average score was 2.89. 74% more or less agreed with the statement; 20% agreed fully.

Particularly after four-way conversations, there is a clear indication that users see the functionality offered by TopicMessenger as more useful than the functionality offered by StandardMessenger.

A clear majority at least partly believes that topic functionality would be a useful addition to their instant messaging program and that it would have a positive impact on their conversations in term of efficiency. However, the users do not think that the functionality will make themselves “more efficient on a day to day basis”.

Despite this, we believe that the data supports the hypothesis; users perceive the topic functionality as useful.

6.8 Users will express an intention to use the topic functionality

We hypothesised that users would express an intention to make use of the topic functionality, if it was made available in their instant messaging client. To assess the users' intentions, they were asked to rate one statement immediately after each conversation and two statements at the end of the experiment on a scale from 1 to 7, where 1 is “Agree” and 7 is “Disagree”. In addition, they were asked to compare StandardMessenger and TopicMessenger directly at the end of the experiment, having used both.

As the questions from the end of the experiment are more directly related to the hypothesis, those are our focus, but the results from the post-conversation comparative questions are included also.

6.8.1 After two-way conversations

	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, pairs level
I would use the messenger for conversations like this one	3.86 (1.78)	3.89 (1.05)	t = 0.05 p = 0.48

Table 6.14: Users' intention to use StandardMessenger or TopicMessenger after two-way conversations.

Users display very similar attitudes towards StandardMessenger and TopicMessenger after two-way conversations.

6.8.2 After four-way conversations

	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, session level
I would use the messenger for conversations like this one	5.1 (0.62)	3.09 (0.71)	t = -4.05 p = 0.002

Table 6.15: Users' intention to use StandardMessenger or TopicMessenger after four-way conversations.

After four-way conversations, users are clearly more negative to using StandardMessenger as compared to TopicMessenger.

6.8.3 At the end of the experiment

These results are also available as charts in Appendix B.

	Mean	SD	% more or less agree		% fully agree	
			Users	Sessions	Users	Sesssions
1. Will use if it becomes available	2.77	1.44	82%	83%	23%	0%
2. Would use it if it was available	2.68	1.39				

Table 6.16: Users' intention to make use of the topic functionality at the end of the experiment.

With average scores of 2.74 and 2.66 and standard deviations of 1.46 and 1.41, the results for these two statements are very similar. In the following, we use the average result for the two.

8 users, or 23% of the experiment participants, are completely in favour of making use of the new features. 82% of the users are more or less favourable to the idea of using the topic functionality. Thus, a clear majority of the users express an intention to make use of the topic functionality, and the hypothesis is supported.

To broaden our understanding of users' attitudes towards topic functionality, they were also asked to make a direct comparison between StandardMessenger and TopicMessenger by answering the question “Which program would you use on a day to day basis?” by circling a number from 1 to 7, where 1 is TopicMessenger and 7 is StandardMessenger.

The percentage who “more or less choose” SM or TM include the participants who circled 1, 2 or 3 versus 5, 6 or 7 respectively. The percentages who chose a client without reservation are those who circled 1 or 7 in the questionnaire.

	Mean	SD	% choose TM without reservation	% more or less choose TM	% more or less choose SM	% choose SM without reservation
"Which program would you use on a day to day basis?"	3.69	2.04	23%	49%	31%	9%

Table 6.17: Users' choice of prototype on a day to day basis.

The numbers reveal a relatively polarised group of users. At 2.04, the standard deviation is the highest among the questions asking users to choose between TopicMessenger and StandardMessenger. Though TM enjoy the largest support, the majority of users are neutral or prefer StandardMessenger. If we aggregate the data to the session level (not shown in the table), 56% of the sessions prefer StandardMessenger.

6.9 Users will perceive the topic functionality as easy to use

To test the hypothesis, users answered questions after each conversation and at the end of the experiment. One question was asked after each conversation, two in the final questionnaire and the users were asked to compare StandardMessenger and TopicMessenger directly.

As the final four questions relate to topic c functionality in general, these are our focus, but the comparative ratings of TopicMessenger and StandardMessenger are also included.

6.9.1 After two-way conversations

	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, pairs level
The messenger was easy to use	1.33 (0.45)	1.5 (0.49)	t = 1.19 p = 0.13

Table 6.18: Perceived ease of use after two-way StandardMessenger and TopicMessenger conversations.

The data shows that experiment participants perceive both messengers as easy to use in two-way conversations, but favour StandardMessenger slightly.

6.9.2 After four-way conversations

	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, session level
The messenger was easy to use	1.74 (0.53)	2.08 (0.68)	t = 1.47 p = 0.09

Table 6.19: Perceived ease of use after four-way StandardMessenger and TopicMessenger conversations.

After four-way conversations, users are less positive to the ease of use of the messengers, but the gap between StandardMessenger and TopicMessenger is only slightly increased.

6.9.3 At the end of the experiment

These results are also available as charts in Appendix B.

	Mean	SD	% more or less agree		% fully agree	
			Users	Sessions	Users	Sessions
1. Topic functionality is clear and understandable	1.647	0.812	97%	100%	53%	11%
2. Using it does not require a lot of mental effort	2.412	1.579	76%	78%	38%	11%
3. It is easy to use	1.941	1.153	91%	100%	41%	0%
4. It is easy to understand how it works	1.412	0.609	100%	100%	65%	11%

Table 6.20: Perceived topic functionality ease of use at the end of the experiment.

On all four criteria, a majority of at least 75% thought the topic functionality was easy to use and understand. On the specific statement "The topic functionality in TopicMessenger is easy to use", 91% agreed to a degree or another and 41% agreed fully. Combined with the post-conversation questionnaire results, the data fully supports the hypothesis.

Although not directly relevant to the hypothesis, users were also asked to make a comparison between TopicMessenger and StandardMessenger.

	Mean	SD	% choose TM without reservation	% more or less choose TM	% neutral	% more or less choose SM	% choose SM without reservation
"Which program was the easiest to use?"	4.46	1.48	6%	11%	57%	32%	14%

On the session level (not shown in the table), 55% preferred StandardMessenger, partially or fully. It is clear that, although TopicMessenger was easy to use, the standard client was perceived as easier to use on average. More than half of the users replied neutrally on this question.

6.10 Users' comments to TopicMessenger

Users had several criticisms of the TopicMessenger client which they raised in the open-ended group interview at the end of the experiment. Though the questions listed in the experiment design section were always covered in the interview, the conversation would often branch out in various directions.

The comments from users included here are available in the original language in Appendix C.

6.10.1 Navigating between topic panes

> *"My problem was that I forgot to switch windows, several times [...] It's probably just a matter of getting used to it, I'm not used to having to switch windows in a conversation" - P6 00:52*

For navigating between topic panes, TopicMessenger lets the user either click on the pane (s)he wants to move to or use the key combinations CTRL + SHIFT + LEFT or RIGHT to move from one pane to the one adjacent. Users commented that it would have been better to use just one keystroke to switch panes.

Also, a user commented that CTRL + SHIFT + LEFT or RIGHT is commonly used in text editing software to select text one word at a time, and that using ALT + LEFT or RIGHT would be a better solution, avoiding the "taken" keyboard shortcuts and sharing the shortcut used by the IM client Adium[12] to switch between chat windows. CTRL+SHIFT+TAB is also a shortcut commonly used to switch between the windows of an application in Windows.

The lesson is to pay close attention to existing design conventions in user interface design, and stick as close as you can to what the user already knows.

6.10.2 Managing topic panes

> *"If someone's chatting in one window and you are typing in another, the stuff they write disappear as you're typing in your window" - P7 13:00*

According to the questionnaire data, the majority of users did not think the topic functionality required a lot of mental effort to use, but several users still commented that keeping track of multiple active chat panes required a conscious effort on their part.

> *"I used more mental capacity to check that I was in the correct window [...] Keeping track of the topic was easier when everyone was in one window" - P11, 04:00*

One user said he wanted to be able to minimize the topic panes and have just one pane visible at a time and use tabs to switch between them, similar to what the OS X messaging client Adium offers users to manage different conversations (see Figure 6.5).

Some users expressed annoyance with the fact that they could not control whether or not topic panes appeared, and would have liked to have the option of ignoring new topic panes, or closing them permanently (in the prototype, panes would re-appear whenever another user wrote something in that pane in his/her client).

> *"When I grew tired of a window [and closed it], it just kept coming back." - P29, 03:20*

[12] Adium: http://www.adiumx.com/ (Retrieved April 20, 2007)

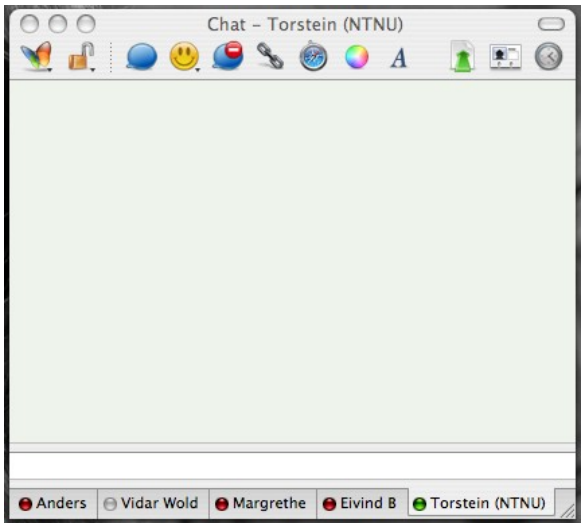


Figure 6.5: The chat window of Adium, a messaging client for OS X.

Considering the feedback, any further development of the prototype should include more robust functionality for managing topics, as discussed in 3.4.4 Lack of user control over topics.

6.10.3 Managing messages

Several users misplaced statements a few times, and would have liked to have the option to move statements from one topic to another. One user suggested a drag and drop interface.

> *"Maybe you should have had a feature to let you move a statement which is in the wrong box over to a different box, because [misplaced statements] do happen, at least with me" - P14, 04:10*

Several users felt it was difficult to keep track of all the new messages appearing in the topic panes they weren't paying attention to. P7 mentioned that color coding, highlighting new messages in other topics until they had been paid attention to, would have been useful.

> *"[It] would be nice with some form of colour coding to highlight new statements in windows I'm not active in, which would go away when I focused on that window" - P7, 05:01*

> *"It can be difficult to see when new messages arrive [when there are many windows]" - P34, 03:10*

This colour highlighting feature was implemented, but intended for the unused threading version of StandardMessenger, and not enabled in TopicMessenger.

The problems users had with tracking the activity in multiple panes echoes comments from users in Grinter and Palen's study (2004), who felt they had a personal threshold for the number of different chat windows they were able to handle at once.

Having to scroll upwards in the chat panes to see what had been going on was mentioned by several users, and provoked comments about the layout of the chat pane:

"Could've been smaller spaces between the messages... I had to scroll up and down a lot, to see what had been said since I checked that window previously" - P7, 04:38

"The brainstorming part of the assignment was OK, but getting decisions wasn't that easy... you had to scroll up to see what people have been doing" - P8, 06:24

This echoes remarks from users in the study by Vronay et al. (1999), where users expressed a wish for a whiteboard or some other persistent medium to make it easer to track the conversation.

6.10.4 The experiment design and questionnaire

"The assignments were realistic [...] If I'm having a conversation, it is because I have something specific to say or ask about" - P8, 00:42

Most users described the assignments they were given as useful starting points, though some paid little heed to them in the conversation, branching off into other subjects quickly.

A user commented that the questions on "efficiency" in the questionnaire were difficult to answer – that what's efficient depends on what you want to achieve with the conversation. Did a decision have to be made here to make the conversation efficient? Or would having a satisfying conversation be sufficient? More guidance to the questions given might have been useful.

6.10.5 "Would you use this functionality in real life?"

"It's not something I've been missing, but if I had access to it I would probably have used it occasionally" - P13, 00:54

The majority of users described the topic functionality as something that would have been a nice but not essential addition to their normal messenger program. We believe this is in line with the data from the questionnaire, where the users show a skew towards TopicMessenger compared to StandardMessenger in terms of perceived conversation quality.

When asked if they could picture scenarios in their day to day lives where the topic functionality would have been useful, several users described more goal-oriented areas of use:

"When we work on school assignments in software engineering, we have more subjects [to discuss] at the same time, and in those situations it would be nice to have it [topic functionality] in the Messenger" - P15, 01:26

"Works best with specific tasks... I normally use IM just to say "hello" and stuff like that, and then it wouldn't feel that natural to split the conversation into different topics" - P5, 02:51

Other users said that using topics made the conversation feel more "formal", and believed it was more likely that the topic functionality would be used in a work context than in a leisure context.

“I think I would have made more use of this functionality in a work context than in my spare time because I rarely separate topics in that way then...” - P10, 05:00 “...yeah, I agree, in more formal situations” - P11

“In professional situations it [topic functionality] would've been nice, but I can't imagine using it when just talking to buddies” - P15 02:00

Another point, made several times, was that topic support felt unnecessary with only two people in the conversation.

“Usually there are no more than two topics, if I want something and the other party wants something, we will in any event be able to keep track of the conversation, mentally” - P8 03:09

“It works best when there are many participants... feels sort of unnecessary with just two people in the conversation” - P6, 02:27

“I would use it primarily in scenarios where I talk to several people [...]” - P32, 05:30

“I think the issue is with the number of people, not with the number of topics... that's when this [explicit topic windows] becomes useful [general agreement]” - P8, 05:15

“In situations with 3 and 4 and not just 2 people, and when you have to have a specific outcome, this could be useful” - P14 02:35

The general consensus among participants that the functionality is best suited to conversations with more than two people. This fits well with the numbers from the log analysis, where four-way conversations were markedly more affected by the introduction of topic functionality in terms of interruptions and users' perceived quality of conversation.

In response to a user commenting that several panes meant that you would have to make more of an effort to keep track of them, another user pointed out that several topics also lets you pay less attention to topics you are less interested in:

“If you enter a box where the topic is 'dinner', well, you know what it's about, you don't have to make that much of an effort [to pay attention]... you know what it's about” P14 04:00

Creating a topic was also seen as a way to draw attention to a specific subject, perhaps to make sure that it is discussed:

“When you talk to somebody, who you don't talk to very often, and you have something important to discuss in addition to talking about bits and bobs, it could be OK to create a topic for that specific thing” -P14, 07:58

7 Discussion

In this chapter, we will discuss the results from the experiment. Results related to the number of interruptions and conversation quality will be discussed first, followed by the user acceptance results. In section 7.9, we consider the low amount of repair statements and apparent lack of confusion seen in the conversations, before we conclude with a section on the validity of the experiment.

7.1 Overall number of interruptions

We hypothesised that the number of interruptions overall would be smaller in TopicMessenger conversations compared to StandardMessenger conversations, since the number of topic interruptions should be reduced and the number of thread interruptions remain stable. The data supported our hypothesis, with 1.527 interruptions per minute in two-way StandardMessenger conversations versus 0.865 in TopicMessenger conversations ($p < 0.001$) and 4.133 interruptions versus 1.891 interruptions per minute in four-way conversations ($p < 0.001$).

From these results alone, we cannot say if the difference was caused by a variation in the number of thread or topic interruptions, but they do show that TopicMessenger somehow reduced the intermingling of unrelated messages.

7.2 Number of topic interruptions

We hypothesised that using TopicMessenger would result in a reduction in the number of interruptions on the topic level. The data supported the hypothesis. For two-way conversations, the average number of interruptions dropped from 0.596 to 0.129 when moving to TopicMessenger ($p < 0.001$). The corresponding values for four-way conversations were 1.126 and 0.215 ($p < 0.001$).

The result shows that TopicMessenger had the intended effect – a clear reduction in the number of topic interruptions. Combined with the results from the user acceptance questions, this demonstrates that users understood the topic functionality and were able to make use of it.

7.3 Number of thread interruptions

We hypothesised that the number of thread interruptions would remain about the same, reasoning that there was no mechanism to help prevent thread interruptions in TopicMessenger. This hypothesis was not supported for four-way conversations – the data indicated a clear decrease in the number of thread interruptions in TopicMessenger in addition to the decrease in topic interruptions.

For two-way conversations, SM conversations had 0.939 interruptions per minute against 0.737 for TM conversations – a difference, but not statistically significant ($t = -1.586$, $p = 0.13$). However, the corresponding values for four-way conversations were 2.872 and 1.675 interruptions per minute, and significant ($t = -4.358$, $p = 0.002$).

Why would the number of thread interruptions decrease markedly in four-way conversations, and not in two-way conversations?

Investigating possible reasons for the difference in thread interruptions between StandardMessenger and TopicMessenger, we compared the intensity in the conversations. We found that both SM and TM conversations increased from about 7 messages per minute to 10.5 messages per minute.

Considering that the intensity of the conversations is about the same, we speculate that the reason for the reduction in thread interruptions is that TopicMessenger lets different users focus on different chat panes. Conversations in StandardMessenger could become quite chaotic at times, with all users exchanging opinions on different subjects in one topic pane; sometimes every statement constituted an interruption. The face to face eqiuvalent would be uncoordinated babble.

Consequently, when a user sent a message on a certain topic, it would prompt the three other users to respond at the same time, and their responses would relate directly to that first message and not to each other, thereby creating “interruptions”. In TopicMessenger conversations, often one or more users would be focusing some attention in a different pane, thereby increasing the time before they reacted to the posted message and making it less likely that three users could create a response and post it simulatenously. This would increase the probability that the sequence of statements following the statement is actually a sequence of statements relating to another, as in a chain, and not just a statement with an assortment of responses.

The data on conversation intensity supports this argument, as there is a clearer association between the intensity of the conversations and the number of interruptions in StandardMessenger conversations as compared to TopicMessenger conversations, indicating that TopicMessenger conversations can have higher intensities without necessarily seeing an increase in interruptions.

Careful further study of the log data could provide more insight into this question, but ultimately it would be difficult to ascertain the user's focus at any given moment through the log data alone. Different sorts of data, for instance of eye movements, would probably be useful.

7.4 Phantom adjacency pairs and repair statements

We hypothesised that the number of phantom adjacency pairs would be lower in TopicMessenger compared to StandardMessenger. We observed very few phantom adjacency pairs in our experiment, and those that did occur did not cause notable confusion. Our results here, combined with those of O'Neill and Martin (2003) and Smith et al. (2000) indicate that phantom adjacency pairs do not represent a significant problem during this type of conversation.

We also hypothesised that fewer repair statements would be found in the conversations conducted in TopicMessenger compared to StandardMessenger. We could not draw any meaningful conclusions as to whether StandardMessenger or TopicMessenger conversations suffer fewer repair statements from the data, since just eight repair statements were found overall during the experiment.

A repair statement is typically an attempt to clear up confusion caused by phantom adjacency statements. Considering the small number of phantom adjacency statements, it is not surprising that there are few repair statements.

In conversation logs, there were very few candidates for repair statements, despite the frequent occurrence of multi-threading (as demonstrated by the data on interruptions). In other words, we could not see any evidence of any phenomenon that regularly caused confusion. This is in line with the findings of O'Neill and Martin (2003), who wrote that users had "little trouble" coping with the multi-threadedness of text chat. The contrast to Garcia and Jacobs' study (1999) is interesting. We discuss this further in section 7.9, "When is multi-threading a problem?".

7.5 Conversations will be easier to follow in TopicMessenger

We hypothesised that the participants would rate TopicMessenger conversations as easier to follow compared to StandardMessenger conversations. The hypothesis was only partly supported. For both two-way and four-way conversations, participants perceived significantly lower level of interruptions in the TopicMessenger conversations. However, with regards to maintaining an overview of the conversation and the perceived coherence of the conversation, a significant difference between the prototypes was found for four-way conversations only,

Interestingly, these results correspond with the number of thread interruptions found in StandardMessenger and TopicMessenger conversations, where two-way conversations saw on average 0.939 and 0.737 thread interruptions per minute, respectively, while four-way conversations suffered 2.872 and 1.675. The 42% difference between the four-way numbers illustrates that TopicMessenger had a larger impact on four-way conversations.

A similar phenomenon was seen when comparing users' assessment of the functionality of the respective clients in 6.7.1 and 6.7.2, whether their functionality "contributed to making the conversation more efficient". The advantage to TopicMessenger is larger in four-way conversations.

We have also seen that several participants stated during the group interview that they felt topic functionality was only really needed for conversations with more than two people. Taken together, the data indicates that topic functionality is less useful for two-way conversations, even with multiple topics, as users are able to handle several conversational threads manually at that scale.

However, in conversations with more participants and several topics being discussed in parallel, the topic functionality helps reduce the intermingling of threads and results in increased conversation quality.

7.6 Users will perceive the topic functionality as useful

We hypothesised that users would perceive the topic functionality as useful. The data largely supported our hypothesis; participants thought the topic functionality was useful and would make conversations more efficient and productive.

Participants did, however, not believe topic functionality would make themselves more efficient. We believe this is a consequence of the leisure focus of the experiment, as well as the role instant messaging usually plays in the everyday lives of the participants. Becoming

more efficient is not a focus when instant messaging for leisure purposes, and the impact of having topic support in the messenger software would probably not be that large seen in the context of a typical user's day.

7.7 Users will express an intention to use the topic functionality

We hypothesised that users would express an intention to make use of the topic functionality, if it was made available to them, and the data supported our hypothesis, with 82% saying the would make use of the functionality.

Although TopicMessenger enjoyed more support than StandardMessenger from users when asked which of the two clients they would use on a day to day basis, the difference was not significant. Considering that a majority of users, 82%, were more or less positive to making use of topic functionality if it becomes available in their instant messaging client, it is interesting that just 48% favour TopicMessenger over StandardMessenger to some degree.

When rating the clients immediately after conversations, when asked whether they would use a client similar to the prototype for conversations such as the one they had just taken part in, users were on average indifferent after two-way conversations (the average rating was 3.86 for SM and 3.89 for TM on a scale where 4 is a neutral response). In four-way conversations, users were markedly more positive to TopicMessenger (average rating: 3.09) and markedly less positive to StandardMessenger (average rating: 5.1).

We believe this partly indicates that users would choose neither StandardMessenger nor TopicMessenger for their daily use, which makes sense, since these clients are very feature-poor compared to commercial clients. However, they would be positive to having topic functionality added into the instant messenger program they are already using.

It is also worth noting that several participants stated that they felt topic functionality was more appropriate for structured contexts, such as work or a school assignment, than leisure chat. We believe this opinion also influenced the mixed attitude observed when participants were asked to choose StandardMessenger or TopicMessenger for daily use.

Finally, according to Davis and Venkatesh (2004), the perceived usefulness of a system is a strong predictor of the intention to use it. The users' clear perception of the topic functionality as useful thus supports the idea that it would be used.

In sum, we believe the data on users' intention to use the functionality and perceived usefulness of the functionality is a clear argument in favour of including topic functionality in commercial messaging clients.

7.8 Users will perceive the topic functionality as easy to use

We hypothesised that users would perceive the topic functionality as easy to use. The data strongly supports our hypothesis, and indicates that we succeeded in creating a graphical user interface that is easy to understand and use.

Some users remarked during the group interview that keeping track of activity in multiple topic panes was demanding. Similar remarks were made by participants in the Flow Chat

study by Vronay et al. (1999) and the Threaded Chat study by Smith et al. (2000). However, given the conversation quality ratings and perceived ease of use score, it appears that these issues were not serious enough to impact the usability of the application.

7.9 When is multi-threading a problem?

While Garcia and Jacobs (1998), Smith et al. (2000), Voida et al. (2002) and Vronay et al. (1999) point out problems caused by multi-threaded conversations, neither the conversations in this experiment nor those of O'Neill and Martin (2003) show any sign of significant confusion arising from the multi-threadedness of the conversations conducted in the chat software.

Conversations conducted in StandardMessenger were heavily multi-threaded, but the phantom adjacency pairs and corresponding confusion among users reported by Garcia and Jacobs did not occur. The heavy multi-threading meant that statements were frequently out of sequence, but this did not seem to bother the users.

Participants did report that they found the conversation in TopicMessenger more coherent than the one in StandardMessenger, but neither chat logs nor the conversation afterwards indicated any significant confusion arising from the fact that the conversations were multi-threaded.

The issue users described as most problematic was the difficulty in tracking all the statements in the four-way conversations, because of a large volume of traffic, and this problem existed both in StandardMessenger and TopicMessenger. Judging by the post-experiment conversations, users' opinion varied: Some users thought separating the different topics made it easier to keep an overview of the conversation or made it possible to purposely ignore parts of it, another felt keeping track of different panes was more mentally demanding than tracking just one pane, even if that pane had correspondingly more traffic.

Why do users in this experiment as well as those observed by O'Neill and Martin (2003) exhibit few signs of confusion compared with those observed by Garcia and Jacobs (1998) and Voida et al. (2002)? We believe the passage of time and the increased prevalence of computer-mediated communication is a factor. Garcia and Jacobs' study is from 1998, and students participating their study had been using the chat programs for eight weeks. We do not know how much previous chat experience they had.

The participants in this experiment had, on average, 7.7 years of instant messaging experience, and conducted an average of 24 instant messaging conversations per week. In other words, they had quite a lot of instant messaging experience. Instant messaging is more widely used today than ever before (America Online 2005, DagensIT 2006). The share of Norwegians aged 16 to 24 years who use the Internet on an average day is up from 16% in 1998 to 80% in 2006. In a poll targeting all adults in the United States, 35% said they went online in 1998. In 2006, this figure had increased to 77%. This makes it increasingly likely that students will be decent typists and be comfortable with computer-mediated communication.

We believe their experience in using computer-mediated communication and instant messaging in particular is the main reason why the participants in this experiment and O'Neill and Martin's study experience few misunderstandings in their instant messaging conversations.

Another example of users coping with a mixture of relevant and irrelevant turns can be found in Herring (1999), where the author suggests that users have grown accustomed to the phenomenon.

7.10 Experiment validity and criticisms

7.10.1 Experiment scale and participant selection

It is important to note that this experiment has a small number of participants, 35, and that these participants are not randomly drawn from the general instant messaging population, but self-selected.

Because of these limitations, we should keep in mind that the conclusions we draw from our data cannot unconditionally be regarded as valid for the instant messaging populace as a whole without further, more extensive research. However, we do believe that our findings can serve as a preliminary indication of the usefulness of topic separation in the chat interface.

7.10.2 Intensity in the experiment compared to other studies

Is the intensity in the conversations in our experiment realistic? Isaacs et al. (2002b) collected thousands of logs from workplace IM interactions and found that users sent about 2 messages per minute on average (in two-way conversations), with 13.5 words per message on average.

Baron (2005) conducted an analysis of 23 IM conversations collected from university students. It was found that the average number of messages sent per minute was 4, and the average message length was 5.4 words. In other words, Baron's study showed that each user sent on average 22 words per minute, while the figure for Isaacs et al.'s study is 27 words per minute. In Campbell et al. (2003), where 16 two-way conversations were conducted, the average figure for messages per minute per user was 2.7, with about 5.3 words per message, so the words per minute figure was about 14.

The corresponding values for this experiment were about 3.5 messages per minute per user for two-way conversations, and about 2.6 messages per minute for four-way conversations. With words per message-values of approximately 7.1 and 6.5, the words per minute-figure is roughly 25 for two-way conversations and 17 for four-way conversations.

Interestingly, Baron hypothesised that the intensity would be higher if not for the multitasking the students in her study engaged in, since most of the students participating should be capable of typing faster than 22 words per minute. In our experiment, 2-3 participants did open a web browser to get relevant information online, and all participants had a paper sheet with information about the assignment they were given. Users probably spent some time referring back to this information during the conversation (they were encouraged to read it before starting the conversation). However, near 100% of the participants' time, once the conversation had begun, was focused on the IM client. Yet the words per minute-figure we found was comparable to that in Baron's study.

In Campbell et al. (2003), participants also devoted all their attention to the experiment, but still experienced even lower intensity levels. It is possible that the conversation intensity would be higher if the participants all knew each other and the subject matter of the conversation was one they were enthusiastic about. It is also possible that the time spent thinking, waiting for the conversation partners' messages, reading them and composing messages means that the "natural" pacing of an instant messaging conversations is around 2-5 messages per minute, depending on the length of the messages. Campbell et al. (2003) found that only 43% of the time users spent with their IM client was used for message composition, which supports the latter idea. Both factors probably play a part.

We believe that the variations in intensity in these studies are due to a number of factors: The experience of the participants involved, the instant messaging client that was used (none of the aforementioned studies used the same client), the amount of multi-tasking by the participants and the different assignments or lack thereof. Overall, however, we believe that the intensity in our experiment conversations is within the norm for instant messaging conversations.

7.10.3 Possible consequences of singular focus on the chat application

More attention devoted to the conversation could result in less confusion using either client and less incentive to multi-thread. Young real-life IM users commonly multi-task while conducting IM conversations (Grinter and Palen 2002, Baron 2005). The experiment does not include any diversionary activities the participants might engage in, so they will have 100% of their time to devote to the conversation they are taking part in.

It is possible that this increased level of attention could result in less turn-taking confusion which in turn could reduce the positive impact of the topic support in TopicMessenger. For instance, users will not be confused by the virtual simultanenity phenomenon discussed in 2.3.5, where two messages appear to arrive one after the other, but in fact appear almost at the same time. When a user is conducting other tasks, (s)he might not be aware that the two new messages that have arrived in the conversation in fact arrived at the same time, and could interpret them as as an adjacency pair – even though they could belong to different topics. In TopicMessenger, the user would return to find the two messages placed in two different topic panes.

In other words, TopicMessenger might appear less effective at reducing confusion in the conversation under experiment conditions than it actually would be in a real-world setting.

Additionally, Voida et al. (2002) cites a user who prefers threaded conversations, as they reduce the need to wait on the replies from other users. As all the participants in the experiment devoted all their time to the conversation, wait times should be lower than in a real-world setting, and thus the incentive to multi-thread in the conversation smaller.

We have seen a correlation between intensity and the number of interruptions, particularly in StandardMessenger conversations. It is possible that an increase in multitasking could reduce the conversation intensity and thus the potential for interruptions.

7.10.4 Conducting just one conversation at a time

While real-life users often take part in more than one conversation at once (with different partners) (Grinter and Palen 2002), only one conversation takes place in the experiment. This

makes it easier to isolate the consequences of including the topic support, but potentially makes topic support less necessary, since the cognitive load on the users is reduced and they can spend more time manually distinguishing statements into different threads of conversation.

7.10.5 No distinction between heavy and light IM users

Isaacs et al. (2002b) note that the "heavy" IM users (users with more than three conversations per day in their experiment) tended to thread their conversations, whereas "light" users did not. TopicMessenger should thus be more useful for heavy IM users, but we did not attempt to specifically recruit participants who were heavy IM users.

IM experience data was collected in the survey participants completed after the experiment, but attempts to correlate IM experience with preferences for StandardMessenger or TopicMessenger and intention to make use of the functionality showed no correlation.

Experiment participants were asked how many instant messaging conversations they had per week. An obvious flaw in the questionnaire was that the meaning of "conversation" in an IM context was not defined, so this data is less meaningful than it should have been. Even so, after accounting for outliers, the average number of conversations per day is 3.24, above the "heavy" threshold of Isaacs et al. (2002b). 43% of the participants who replied to the question reported more than 21 conversations per week, meaning more than 3 per day. Three participants wrote "Many". As previously noted, the average number of years of IM experience was 7.76. The minumum number of years of experience among the participants was 3.

We did not find any correlation between the participants' level of experience and their attitudes towards TopicMessenger, and we believe that the reason is the long experience with IM the participants in the study had.

7.10.6 Not capturing all the information in the log

As noted by Campbell et al. (2003), the log does not capture all the information about what is happening in the user's interaction with the computer and the other users. In their study, Campbell et al. found that users spent only 43% of their time actually typing, and 12% revising text already entered but not sent.

Screen captures from the participants were collected during the experiment, but full video analysis proved too resource demanding considering that we are not focusing on the potential differences in behaviour during message composition between StandardMessenger and TopicMessenger in this thesis. Detailed analysis of users' behaviour during intensive multi-topic text chat is therefore left to future research.

7.10.7 Unnatural experiment situation

Instant messenger users usually have some natural reason to interact. They might know each other, or they are assembled for some purpose, like the participants in O'Neill and Martin's study (2003) are.

The experiment scenario attempts to give the participants some common ground on which to participate, but it is difficult to replicate a situation where the participants know each other already (Jacobs and Garcia 1999), have just gone through a seminar together (O'Neill and

Martin 2003) or are friends or colleagues (Baron 2005, Isaacs et al. 2002b). In this experiment, conversation flowed naturally or stuck strictly to the assignments the users were given, probably depending on the personalities of the participants.

The experiment conversations were predetermined to last 15 minutes, and while this is unnatural, it is a compromise between a completely free-form experiment and one that is long enough to let the users test the functionality properly. 15 minutes is also more than the average length of conversations found by by Isaacs et al. (2002b), which was 4.4 minutes, but shurter than the average length measured by Baron (2005), 23.7 minutes.

It is inevitable that the experiment situation will affect the results to some degree. However, in the group interview, the participants were asked how they felt about the experiment context, and there were no objections raised. This, the contents of the logs, where the vast majority of users participated actively in the conversations, and the fact that 34 of the 35 participants contributed at least 28 messages in both TopicMessenger and StandardMessenger conversations leads us to believe that the experiment situation was "natural enough" to provoke the type of leisurely conversation we were aiming for.

7.10.8 Topic selection bug

A bug in TopicMessenger made switching topics more difficult than it should be. Specifically, the program detected only mouse button press/button release sequences, not just button press. Therefore, users' clicks on chat panes sometimes would not register, meaning that unless they noticed that the click did not register they would type statements into the wrong pane – misplacing them.

This bug was fixed after the first three sessions, but the consequences of the misplaced statements are included in the data (showing up as topic interruptions). We decided to include this data since it indicates only a small drop in the average number of topic interruptions for sessions 4-9 compared to 1-3. The increased likelihood of topic interruptions occurring in these conversations can only weaken our hypotheses, if it has any effect at all.

In retrospect, the client should have been tested more thoroughly, and in the interest of scientific correctness (testing the exact same client with all users), the error should perhaps have been left in the program. We decided on correcting it because it took attention away from the functionality we wanted to test and thus reduced the amount of useful data that was gathered.

7.10.9 Software instability

During actual usage in the experiment locale, the prototype software proved less stable than during preliminary testing, which was conducted in a different network. A majority of the experiment sessions experienced a breakdown in the commununication between one prototype client and the server at least once. Fortunately, the user could return to the conversation within seconds by disconnecting and connecting, and the state of the chat panes in the client was preserved, so the user could continue where he or she left off.

After the problem became apparent, all participants were asked whether it affected the experiment in the post-experiment open-ended group interview. No participants felt that the

experiment was affected. One participant remarked that he would not have tolerated a client being so unstable in real use, but added that for the purpose of the experiment it made no difference.

8 Future work

In this chapter, we consider possible avenues for future work on topic-enabled chat user interfaces, chat applications in general and logging.

8.1 Alternative usage scenarios for topic functionality

Users did express an intention to use topic functionality if it becomes available in their chat client, and rated four-way conversations in TopicMessenger as easier to follow compared to those from StandardMessenger. In the group interviews following the experiments, there was a clear consensus among the participants that using topic functionality made sense mostly when having more than two people in the conversation. Several participants said topic functionality would be useful to a greater extent in a more formal context, for instance work or school assignments, where there are clear issues that need to be tackled in the conversations.

Compared to Smith et al. (2000) and Vronay et al. (1999), who did not consciously provoke multi-threaded conversation in their experiment, this experiment aimed to provoke multi-threading in conversations by assigning different subjects to different participants and restricting the amount of time available to discuss those subjects. Smith et al. placed their experiment in a work context, and provided the participants with specific tasks. This experiment attempted to create a situation more similar to that faced by users in everyday leisure instant messaging use. Therefore, the participants were told that it was OK to stray from the subjects they were given if they wanted to. Vronay et al. (1999) took a similar approach, but did not provide participants with specific different topics.

Topic functionality might be more valuable in a situation with stricter constraints – more people in the conversation and a set of subjects that must be discussed within a set time limit. We believe this type of situation is found more often in online meetings in business – for instance in software engineering projects. Projects that have cost and equipment constraints (for instance geographically distributed open source projects such as the Health Information Systems Programme[13]) and therefore cannot conduct phone conferences are more likely to conduct these meetings with instant messaging tools.

A project meeting with 5 – 10 participants, a wide range of topics that must be dealt with and participants who have varying levels of time to devote to the chat could be an interesting usage scenario. The large number of participants could mean that some project members have little to say on certain topics but more on others – separating the conversation into multiple topics would mean that they could pay less attention to some topics and more to others. Participants could combine the online meeting with work, and pay less attention to the meeting when activity centred on topics not relevant to them. Returning to the conversation, they can spend their energy on catching up on only those topics where their input is required.

This type of scenario is more in line with what Chen et al. (2005) envisions in IBM's patent application for a topic-enabled instant messaging client, and brings us into the field of computer-supported collaborative work (CSCW), which is outside the scope of this thesis.

Moving towards more structured chats, where a meeting leader designates the next speaker, might be a better solution in terms of keeping the conversation focused, but this slows down

[13] Health Information Systems Programme: http://www.hisp.org/ (Retrieved April 30, 2007)

and changes the nature of the communication, as O'Neill and Martin (2003) remarks. For text chat meetings where discussion is necessary, the reduced speed of the interaction could be a significant objection to this type of structured chat.

8.2 Determine the value of separating topics in more formal situations

In this thesis, the focus is on conversations in a leisure context. As discussed in the previous chapter, our findings have led us to believe that the usefulness of a topic-enabled client would be greater in a more formal situation. This could be tested by recruiting members of an existing project, for instance an open source software engineering project or students participating in project work as a part of their classes. The project members would conduct their meetings using a tool similar to TopicMessenger. The log and survey data from such a session would then be compared to a meeting conducted in the instant messaging client that is normally used.

Any prototype would have to be more properly tested before being used in such a scenario, especially as the network conditions would probably be more difficult with geographically distributed users resulting in more lag. A pilot test would be needed to ensure that all features required by the project members to conduct a meeting is functional. Ideally, the experiment should be repeated with different project groups.

If recruiting a real project for the meeting proves difficult, it is of course possible, but not desirable, to recruit random participants for the experiment. In that scenario, it would be important to recreate a project meeting situation as closely as possible, meaning that different users should have different instructions and preferably have a thorough introduction to the "project" they would be discussing.

8.3 Field trial of full-featured program

The results of the experiment indicate that users would make use of topic functionality if it was integrated into their client, even though they did not envision that they would need it for two-way exchanges with their friends. As Vronay et al. (1999) note, it is difficult to make valid statistical comparisons when users are unfamiliar with the new concepts and compare them to concepts they are very experienced with. To investigate the actual value of such a feature, a more full-featured version of the client could be created, either based on the current client or by modifying a cross-platform, multi-network client such as Pidgin. It would have to be compatible with an existing chat network, such as the Microsoft Network (MSN), so that its users could replace their current client with the topic-enabled prototype.

The prototype would function as a normal messaging client when communicating with standard clients, but when talking to another prototype client, topic functionality would be accessible to the user and the data required to make it work embedded in the messages. Ideally, the client would be freely downloadable and return anonymous usage statistics, providing a more realistic picture of how useful topic functionality is in real life.

In general, it would be interesting to see hybrid clients incorporating both "standard" instant messaging functionality as well as experimental features, while running on established IM networks. In the literature, there are several examples of research laboratories deploying their

chat or instant messaging prototypes widely within their companies (Erickson et al. 1998, Isaacs et al. 2002a). Isaacs et al. make Hubbub available online to learn more about its use in a broader community[14], but it uses its own, separate network. We believe piggybacking on existing networks would increase adoption of prototype clients, since users would still be able to have basic access to their buddies in the prototype client.

Using a prototype which is relying on its own network means that the user will have to run it in parallel with the standard client and find others with the prototype to communicate with; extra hassle which we believe will restrict such new clients' opportunities to flourish. Of course, some features might require new server-side technology to work, even then it would be advantageous for the client to piggyback on existing infrastructure as far as possible and utilize special servers only when necessary for the new functionality to work.

8.4 Increase users' awareness

Vronay et al. (1999) reported subjectively more coherent chat histories in their Status Chat prototype, where users could see what the other participants in the conversation were composing in a separate pane. This functionality meant that users could (and did) cooperate in timing their statements to make more coherent conversations.

It would be interesting to conduct a formal experiment comparing the number of interruptions in a client similar to Status Chat with a baseline client, similar to what we have done with StandardMessenger and TopicMessenger, or even to combine topic functionality with the increased awareness enabled by Status Chat.

8.5 Automatic classification of statements into threads or topics

Chen et al. (2005) outline a method for classifying statements into topics by keyword, thereby potentially reducing the work the user has to perform to make statements appear in the correct topic. Shen et al. (2006) describe algorithms for automatically sorting intermingled chat messages into different conversation threads.

In this thesis, we have focused on the impact of dedicated topics panes in a prototype where the user has to manually select which topic his statements belong to. It would be interesting to investigate the impact of mechanisms that automatically sort statements for the user.

8.6 Determine the impact of attention on the conversation

As discussed in the previous chapter, users devoted all their time to the conversation in this experiment, which is unrealistic. We believe that reducing the amount of attention a user can focus on the conversation will make the conversation complexity (in terms of multi-threading) more difficult to handle for that user, provided a certain minimum level of intensity is present.

In a field trial, the amount of time users spent in the chat pane could be measured by the application, so that users who spend much time multi-tasking can be compared to those who focus solely on the conversation.

[14] Hubbub Info: http://www.hubbubme.com/info.php (Retrieved April 30, 2007)

If an experiment was used to examine this, extra tasks would have to be introduced. For instance, users could be required to go online with a web browser to find information needed in the conversation. This type of distraction was considered for the experiment in this thesis, but was dropped to reduce experiment complexity.

8.7 More sophisticated graphical interfaces

Bartram (1997) discusses the potential of motion in displaying information in user interfaces. If a more full-functioned client was developed, it could be worth investigating the value of motion in the messenger design. For instance, a version of TopicMessenger that supported dragging messages between topic panes (to correct misplaced messages) where the message disappeared from Pane A and appeared in Pane B, perhaps with a colour highlight, could be compared to an enhanced client where the message smoothly slides from Pane A to Pane B.

After the experiment, some users mentioned that tracking the activity in other topic panes than the one they were currently active in required significant mental effort. Klein and Bederson (2005) illustrate the benefits of smooth text scrolling, both in terms of efficiency, error rates for reading tasks and user satisfaction. Vronay et al. (1999) reported that their smooth-scrolling Flow Chat, in which 75% of the screen was moving continuously, induced vertigo among its users. Clearly, a balanced approach to smooth scrolling in instant messaging applications would be worth investigating.

8.8 Logging

In this study, the possible gain from the clearer log data a topic enabled client might yield has been out of focus. Both Chen et al. (2005) and Smith et al. (2000) point out the potential increased value of more organized log data. Storing log data from different topics separately means that users do not have to filter the conversation transcript to remove the messages that hold no useful information to them.

In the TopicMessenger prototype, logs are stored both in the form of text, with all statements in the conversation, irrespective of topic, organized sequentially, and as a binary file that can be played back in the TopicMessenger application. During playback, the application tracked the number of words and messages per user.

The purpose of this functionality was to enable log analysis. Though viewing a playback of a conversation can be useful, it would probably be more important to provide easy access to topic-specific logs. In a professional context, where information must be made available to a wider organization, it would be natural to integrate a desktop client such as TopicMessenger with online repositories, where logs can be stored and read through a web browser, for instance. Ideally, the log repository should be integrated with other knowledge management systems used by the organization, for instance wikis.

Babble (Erickson et al. 1999) is one chat system where conversations persist automatically on the server and are available to users when they join the server. This means that the chat environment takes on some of the persistence of a web forum. Lotus Sametime (IBM Lotus Sametime 2007) is a business oriented messaging system where log transcripts can be automatically saved on the server side.

9 Conclusion

The purpose of this thesis was two-fold. First, to investigate user interfaces for instant messaging conversations to try and find an interface that improves the quality of conversation, compared to standard instant messaging user interfaces. Second, to investigate whether users express a perceived intention to use the new functionality offered by such a user interface.

To this end, we have conducted a survey of instant messaging research literature, prototyped different approaches to chat interface design and compared one of the prototypes to a standard user interface in an experiment with 35 students.

The existing research points out several problems of text chat, of which confusion arising from the intermingling of conversation threads (Garcia et al. 1998, Voida et al. 2002, Vronay et al. 1999) is the main focus of this thesis. By creating a chat client, TopicMessenger, where users can create separate panes for each topic in the conversation, we hoped to reduce the problems caused by multi-threading in chat conversations. TopicMessenger was compared to a similar prototype without the topic support, StandardMessenger, in a series of leisure-themed conversations with 2 or 4 participants. Data was gathered through log analysis, questionnaires and open-ended group interviews.

Our two main hypotheses were that conversation quality would be improved in TopicMessenger conversations compared to StandardMessenger conversations, and that users would express a high degree of acceptance for TopicMessenger,

The results from the log analysis show that compared to the standard chat interface, the amount of multi-threading is significantly lower in TopicMessenger. Users perceived the number of interruptions as smaller and the coherence of the conversation as better in four-way conversations in TopicMessenger compared to StandardMessenger. Overall, a clear majority of users rated conversations in TopicMessenger as more coherent and with fewer interruptions than conversations in StandardMessenger.

A clear majority of users perceived the topic functionality in TopicMessenger as easy to use (91%) and useful (74%), and believed they would make use of the functionality if it was included in their instant messaging client (82%). Both hypotheses were thus supported.

Although this is a clear indication that topic functionality has benefits for some users, the small sample of self-selected participants in this experiment means that further research is needed to validate our findings.

Through the open-ended group interviews conducted after the experiment, it emerged that many participants saw the topic functionality as better suited to more formal or structured conversations and unnecessary in two-way leisure-oriented conversations. A natural direction for future research would be to investigate the value of dedicated topic structures in more formal contexts, such as online project meetings.

Several experiment participants commented that tracking conversation threads at multiple locations on-screen simultaneously was demanding. There are a number of possible user interface enhancements, such as smooth state transitions and scrolling, which could help make such tasks easier.

Finally, considering that the majority of the users in this study expressed an intention to use topic functionality if it becomes available in their instant messaging clients, this type of conversation structuring aids should be considered for inclusion in consumer-oriented instant messaging clients.

10 References

America Online, 2005, "Third Annual Instant Messaging Survey". Retrieved March 30, 2007, from http://www.aim.com/survey/

Baron, N. S., 2005, "Instant Messaging by American College Students", Retrieved April 10, 2007, from http://www.american.edu/tesol/Baron-AAAS-IM%20by%20American%20College%20Students.pdf

Bartle, R., 1990, "Early MUD History", Retrieved April 16, 2007, from http://www.lud-d.luth.se/mud/aber/mud-history.html

Bartram, L., 1997, "Can Motion Increase User Interface Bandwidth in Complex Systems?", 1997, *"Computational Cybernetics and Simulation", IEEE International Conference on Systems, Man, and Cybernetics,* vol. 2, pp. 1686-1692.

Bridge, P. D., Sawilowsky, S. S., 1999, "Increasing Physicians' Awareness of the Impact of Statistics on Research Outcomes: Comparative Power of the t-test and Wilcoxon Rank-Sum Test in Small Samples Applied Research", *Journal of Clinical Epidemiology,* vol. 52, no. 3, pp. 229-235.

Cameron, A. F., Webster, J., 2004, "Unintended consequences of emerging communications technologies: Instant Messaging in the workplace", *Computers in Human Behavior,* vol. 21, pp. 85-103.

Campbell, J. D., Stanziola, E., Feng, J., 2003, "Instant Messaging: Between the Messages", *2003 IEEE International Conference on Systems, Man and Cybernetics, v*ol. 3, pp. 2193-2198.

Chatterjee, S., Abhichandani, T., Haiqing, L., TuIu, B. Jongbok B., 2005, "Instant messaging and presence technologies for college campuses", *IEEE Network,* vol. 19, issue 3, pp. 4-13.

Chen, Y., Handy-Bosma, J. H., Selvage, M. Y., Walker, K. R., 2005, "System and method for in-context, topic-oriented instant messaging", Patent US/2005/0262199/A1.

ComScore, 2006, "Europe Surpasses North America in Instant Messenger Users, comScore Study Reveals". Retrieved April 12, 2007, from http://www.comscore.com/press/release.asp?press=800

Creswell, J. W., 2003, *Research Design: Qualitative, Quantitative and Mixed Methods Approaches,* 2nd edn, Sage Publications, Thousand Oaks.

Curtis, P., 1992, "Mudding: Social Phenomena in Text-Based Virtual Realities", *Conference on Directions and Implications of Advanced Computing*, pp. 1-21.

DagensIT, 2006, "Lynmeldinger er jokeren". Retrieved January 30, 2007, from http://www.dagensit.no/esc/article696489.ece

Davis, F. D., Venkatesh, V., 2004, "Toward Preprototype User Acceptance Testing of New Information Systems: Implications for Software Project Management", *IEEE Transactions on Engineering Management,* vol. 51, no. 1, pp. 31-46.

Deitel, H. M., Deitel, P. J., 2001, *Java: How to program*, 4th edn, Prentice Hall, Upper Saddle River, New Jersey.

Dix, A. J., Finlay, J., Abowd, G. D., and Beale, R., 2004, *Human-Computer Interaction*, 3rd edn, Pearson, Prentice Hall, Harlow, England.

Ducheneaut, N., Moore, R. J., 2004, "The Social Side of Gaming: A Study of Interaction Patterns in a Massively Multiplayer Online Game", *CSCW '04*, vol. 6, issue 3, pp. 360-369.

EFnet 2007, "efnet statistics". Retrieved January 30, 2007, from http://stats.efnet.org/

Furnas, G. W., 2006, "A Fisheye Follow-up: Further Reflections on Focus + Context", *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pp. 999-1008.

Erickson, T., Smith, D. N., Kellogg, W. A., Laff, M., Richards, J. T., Bradner, E., 1999, "Socially Translucent Systems: Social Proxies, Persistent Conversation, and the Design of 'Babble'", *CHI 99*, pp. 72-79.

Garcia, A., Jacobs, J. B., 1998, "The Interactional Organization of Computer Mediated Communication in the College Classroom", *Qualitative Sociology,* vol. 21, no. 3, pp. 299-317.

Gowan, M., 2000, "How it works: Instant messaging". Retrieved January 31, 2007, from http://archives.cnn.com/2000/TECH/computing/05/25/how.messaging.works.idg/index.html

Greene, D., O'Mahoney, D., 2004, "Instant Messaging & Presence Management in Mobile Ad-Hoc Networks". *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*.

Greenwald, A. G., 1976, "Within-subjects Designs: To Use or Not To Use?", *Psychological Bulletin,* vol. 83, no. 2, pp. 314-320.

Grinter, R. E., Palen, L., 2002, "Instant Messaging in Teen Life", *CSCW '02*, pp. 21-30.

Hentschel, E., 1998, "Communication on IRC", *Linguistik online 1/98.* Retrieved April 10, 2007, from http://www.linguistik-online.de/irc.htm

Herbsleb, J. D., Boyer, D. G., Handel, M., Atkins, D. L., Finholt, T. A., 2002, "Introducing Instant Messaging and Chat in the Workplace", *CHI 2002,* vol. 4, issue 1, pp. 171-178.

Herring, S., 1999, "Interactional Coherence in CMC", *Proceedings of the 32nd Hawaii International Conference on System Sciences*.

IBM Lotus Sametime, 2007, "IBM Lotus Sametime – enterprise instant messaging and web conferencing". Retrieved April 5, 2007, from http://www-142.ibm.com/software/swlotus/sametime

Isaacs, E., Walendowski, A., Ranganthan, D., 2002a, "Hubbub: A sound-enhanced mobile instant messenger that supports awareness and opportunistic interactions", *CHI 2002,* pp. 179-186.

Isaacs, E., Walendowski, A., Whittaker, S., Schiano, D. J., Kamm, C., 2002b, "The Character, Functions and Styles of Instant Messaging in the Workplace", *CSCW '02*, pp. 11-20.

Kaufman, R. J., Li, X., 2003, "Payoff Externalities, Informational Cascades and Managerial Incentives: A Theoretical Framework for IT Adoption Herding", *Proceedings of the 2003 INFORMS Conference on IS and Technology*.

Klein, C., Bederson, B. B., 2005, "Benefits of Animated Scrolling", *CHI 2005*, pp. 1965-1968.

Living Internet a, "Early chat programs". Retrieved January 29, 2007, from http://www.livinginternet.com/r/ri_early.htm

Living Internet b, "PLATO". Retrieved January 29, 2007, from http://www.livinginternet.-com/r/ri_talk.htm

Living Internet c, "Compuserve / CB". Retrieved January 29, 2007, from http://www.livinginternet.com/r/ri_compuserve.htm

Lonchamp, J., 2005, "A structured chat framework for distributed educational settings", *Proceedings of the 2005 Conference on Computer Support for Collaborative Learning*, pp. 403-407.

Mathison, S., 1988, "Why Triangulate?", *Educational Researcher,* vol. 17, no. 2, 13-17.

McKean, E. (ed.), 2005, *The New Oxford American Dictionary*, 2nd edn, Oxford University Press.

Microsoft, 2006, "Yahoo! and Microsoft Bridge Global Instant Messaging Communities". Retrieved 28 April, 2007, from http://www.microsoft.com/press-pass/press/2006/jul06/07-12IMInteropPR.mspx

Moore, D. S., McCabe, G. P., 2003, *Introduction to the Practice of Statistics*, 4th edn, W. H. Freeman and Company, New York.

Nardi, B. A., Whittaker S., 2000, "Interaction and Outeraction: Instant Messaging in Action", *CSCW '00*, pp. 79-88.

O'Neill, J., Martin, D., 2003, "Text Chat in Action", *Proceedings of the 2003 International ACM SIGGROUP conference on Supporting group work*, pp. 40-49.

Ogura, K., Ishizaki, M., Nishimoto, K., "A Method of Extracting Topic Threads Towards Facilitating Knowledge Creation in Chat Conversations" in *Knowledge-Based Intelligent Information and Engineering Systems*, pp. 330-336.

Oikarinen, J., "IRC History". Retrieved January 30, 2007, from http://www.ircnet.org/History/jarkko.html

Pew / Internet, 2004, "How Americans use Instant Messaging". Retrieved April 10, 2007, from http://www.pewinternet.org/PPF/r/133/report_display.asp

Preece, J., Rogers, Y., Sharp, H., 2002, *Interaction Design,* John Wiley & Sons, New York.

Schulze, M., 1998, "Substitution of paraverbal and nonverbal cues in the written medium of IRC". Retrieved April 28, 2007, from http://citeseer.ist.psu.edu/664283.html

Shen, D., Yang, Q., Sun, JT., Chen, Z., 2006, "Thread detection in dynamic text message streams", *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval",* pp. 35-42.

Smith, M., Cadiz, JJ., Burkhalter, B., 2000, “Conversation Trees and Threaded Chats”, *Proceedings of the 2000 ACM conference on Computer Supported Cooperative Work*, pp. 97-105.

Tran, M. H., Yang, Y., Raikundalia, G. K., 2005, “Supporting awareness in instant messaging: an empirical study and mechanism design”, *Proceedings of the 19th conference of the computer-human interaction special interest group (CHISIG) of Australia on Computer-human interaction,* pp. 1-10.

Undernet, 2007. Retrieved January 29, 2007, from http://www.undernet.org/

Voida, A., Newstetter, W. C., Mynatt, E. D., 2002, “When conventions collide: the tensions of instant messaging attributed”, *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, pp. 187-194.

Viegas, F. B., Donath, J. S., 1999, “Chat Circles”, *CHI '99*, pp. 9-18.

Vronay, D., Smith, M., Drucker, S., 1999, “Alternative interfaces for chat”, *Proceedings of the 12th annual ACM symposium on User interface software and technology*, pp. 19-26.

Warren, M., 2006, *Features of Naturalness in Conversation*, John Benjamins Publishing Company, Amsterdam, Philadelphia.

Wikipedia, 2007a, “AIM”. Retrieved January 29, 2007, from http://en.wikipedia.org/wiki/AIM

Wikipedia, 2007b, “CB Simulator”. Retrieved January 29, 2007, from http://en.wikipedia.org/wiki/CB_Simulator

Wikipedia, 2007c, “IM Clients”. Retrieved January 31, 2007, from http://en.wikipedia.org/wiki/Comparison_of_instant_messaging_clients

Wikipedia, 2007d, “MSN Messenger”. Retrieved January 30, 2007, from http://en.wikipedia.org/wiki/MSN_Messenger

Wikipedia, 2007e, “Yahoo! Messenger”. Retrieved January 30, 2007, from http://en.wikipedia.org/wiki/Yahoo!_Instant_Messenger

Wikipedia, 2007f, “Games and applications for Windows LIve Messenger”. Retrieved April 22, 2007, from http://en.wikipedia.org/wiki/Games_and_applications_for_Windows_Live_Messenger#Application_Sharing

Wikipedia, 2007g, “Windows Messenger”. Retrieved April 28, 2007, from http://en.wikipedia.org/wiki/Windows_Messenger

Wikipedia, 2007h, “Mirabilis”. Retrieved April 28, 2007, from http://en.wikipedia.org/wiki/Mirabilis_(company)

Wikipedia, 2007i, “mIRC”. Retrieved April 15, 2007, from http://en.wikipedia.org/wiki/Mirc

Wilcoxon, F., 1945, “Individual Comparisons by Ranking Methods”, *Biometrics Bulletin*, pp. 80-83.

Zitzen, M., Stein, D., 2003, “Chat and conversation: A case of transmedial stability?”, *Linguistics,* vol. 42., no. 5, pp. 983-1021.

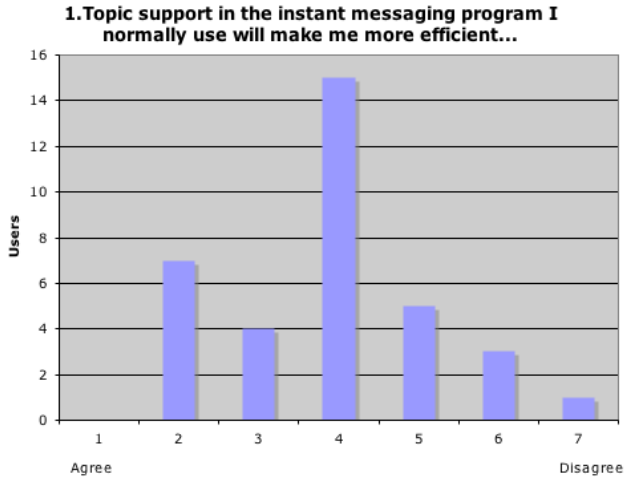
Appendix A – List of abbreviations

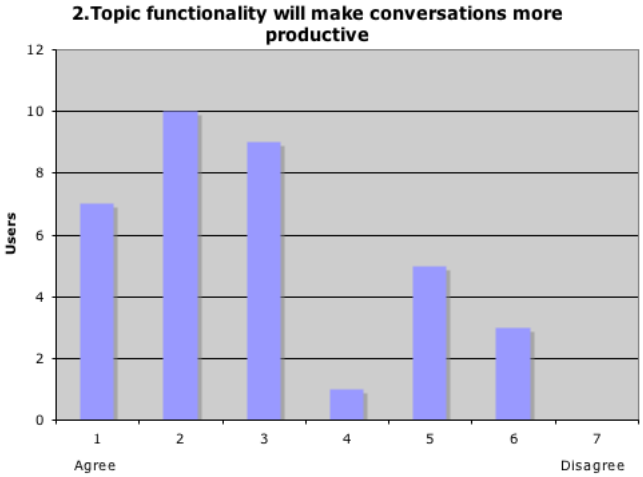
The following is a list of abbreviations and their meaning in this thesis.

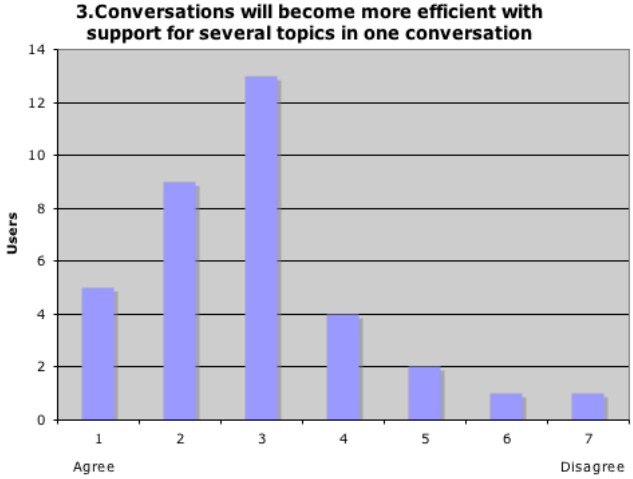
AIM	AOL Instant Messenger
AOL	Formerly America Online, currently just AOL
CMC	Computer-Mediated Communication
CSCW	Computer-Supported Collaborative Work
ICQ	A play on the phrase “I seek you”
IM	Instant Messaging
IRC	Internet Relay Chat
MSN	Microsoft Network
MUD	Multi-User Dungeon
SM	StandardMessenger
TM	TopicMessenger

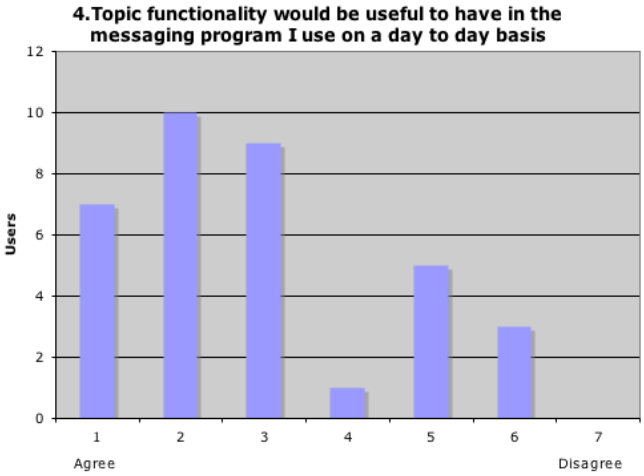
Appendix B – Charts of acceptance data

Users' perceived usefulness of topic functionality

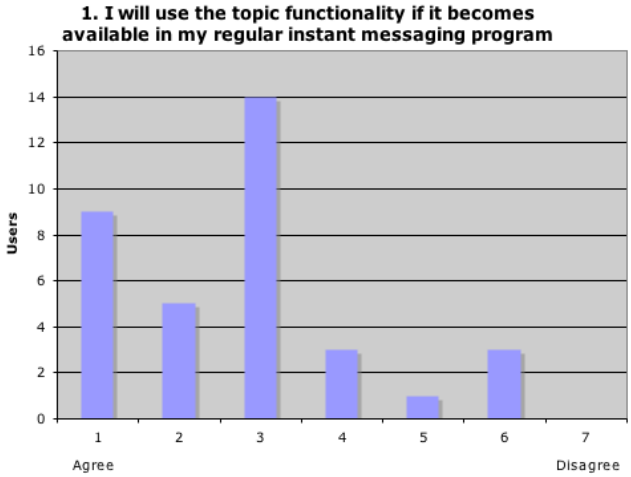


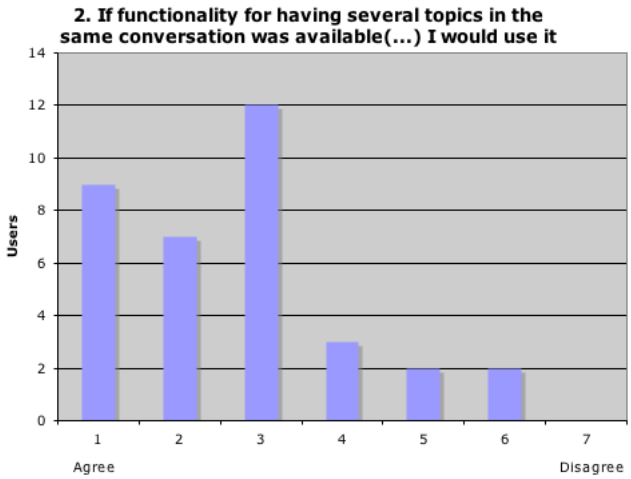


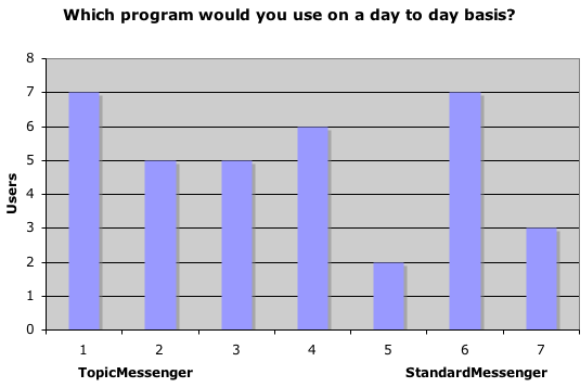




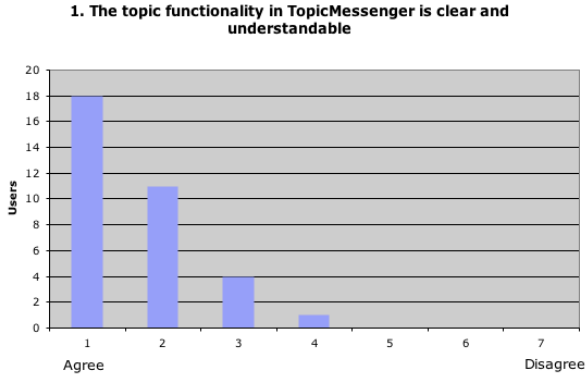
Users' perceived intention to use the topic functionality

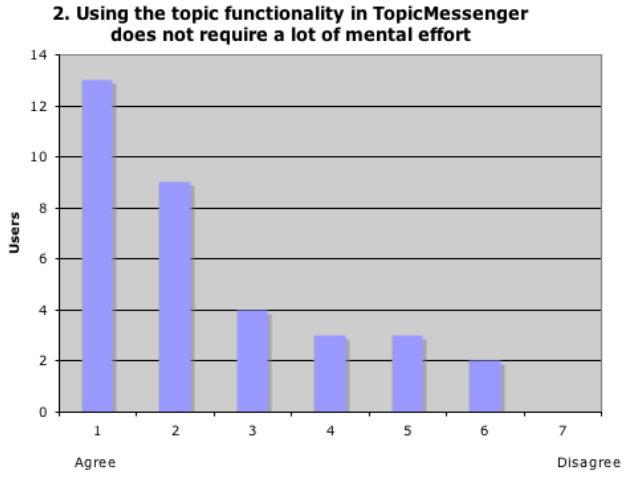


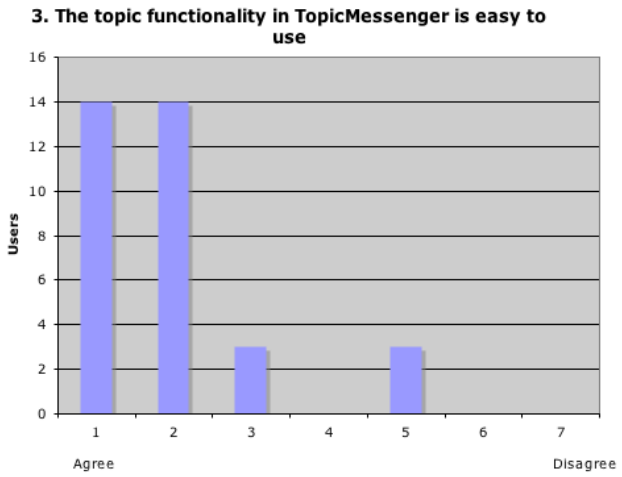


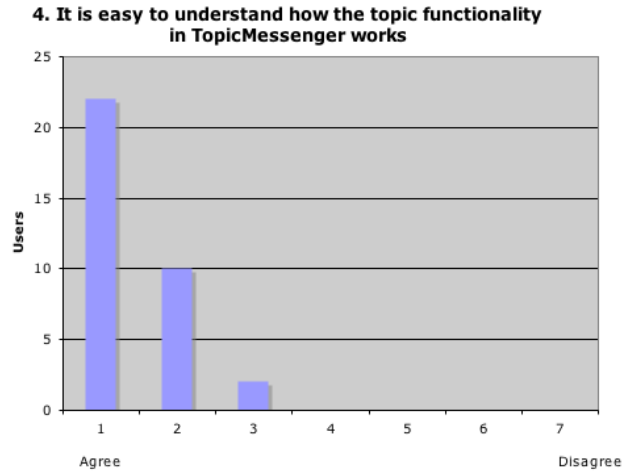


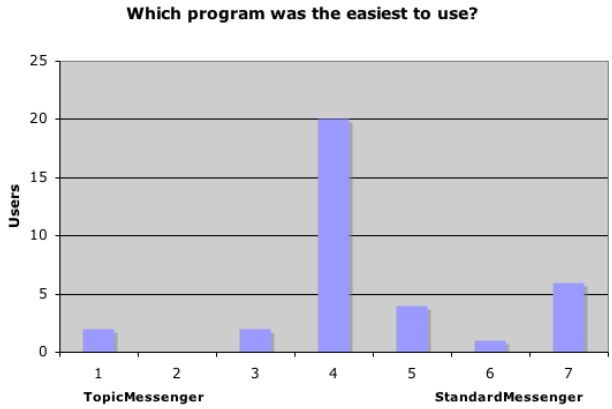
User's perceived ease of use






3. The topic functionality in TopicMessenger is easy to use
Users
16
14
12
10
8
6
4
2
0
1
2
3
4
5
6
7
Agree
Disagree


4. It is easy to understand how the topic functionality in TopicMessenger works
Users
25
20
15
10
5
0
1
2
3
4
5
6
7
Agree
Disagree


Which program was the easiest to use?
Users
25
20
15
10
5
0
1
2
3
4
5
6
7
TopicMessenger
StandardMessenger

Appendix C – Quotes in the original language

The informal conversations at the end of the experiment were conducted in Norwegian. Quotes from these conversations are translated to Norwegian in the main document. Here, they are presented in participant order, from Session 1 through Session 9.

Session 2

“[kan være nyttig] hvis det er noen konkrete temaer man skal diskutere.. jeg bruker det [instant messaging] ikke så mye, men når jeg skal bruke det er det for å si hallo og hei på deg [...] jeg bruker det ikke så mye i hverdagen for å avtale ting [...] og da er det ikke så naturlig å dele det opp i temaer, kanskje” - P5, 02:51

“Det som ikke funket for meg, var at jeg glemte å skifte vinduer, opptil flere ganger.. det er jo bare tilvenning, men det skjer ofte at jeg bare fortsetter i samme vinduet, jeg er jo ikke vant til å måtte gjøre det” - P6,00:52

“Men det funker best når det er mange, da, når det bare er to føles det litt unødvendig, kanskje” - P6, 02:27

“Det var veldig realistisk, det synes jeg i alle fall [...] Typisk, når jeg starter en samtale, gjør jeg det fordi jeg har noe konkret jeg vil si eller spørre om” - P8, 00:42

“Jeg kunne godt tenkt meg at det var litt mindre luft mellom alle personene, altså, for det ble veldig mye scrolling opp og ned... hva har blitt skrevet siden sist jeg tittet i denne ruta?” - P7, 04:38

“[det hadde vært fint å ha ny aktivitet] markert med en eller annen farge, for eksempel, slik at markeringen forsvinner når jeg flytter fokus til det vinduet” -P7, 05:01

“[...] hvis det er noen som chatter i noen kolonner og du prøver å skrive noe fornuftig i en tredje kolonne, så forsvinner jo det andre langt av gårde før du får skrevet en liten setning i et annet vindu” - P7 13:00

“Somoftest blir det ikke mer enn to temaer, hvis jeg vil noe, og den andre vil noe, så klarer vi uansett mentalt å holde orden på samtalen” - P8, 03:09

“Jeg tror at primærproblemet ligger i antall personer, ikke i antall temaer... altså, det er da det [emnefunksjonalitet] blir aktuelt [generell enighet]” - P8, 05:15

“[...] brainstorminga [gikk greit], men å få avgjørelser var ikke like litt hele tiden, det blir veldig mange forslag, [...] og så må man begynne å scrolle opp for å se hva folk egentlig har gjort” - P8, 06:24

Session 2

“Jeg brukte mer mental kapasitet på å sjekke at jeg var i riktig vindu enn...” - P11 “...jeg skrev i feil vindu et par ganger...” - P10 “..ja, det skjedde med nesten alle... det var enklere å finne temaet når alle var i samme vindu” -P11, 04:00

“Jeg tror jeg hadde fått mer bruk for det i jobbsammenheng enn i fritida,fordi jeg pleier ikke dele opp emner på samme måte da” -P10, 05:00 “Ja, det tenkte jeg også... i mer formelle situasjoner” - P11

Session 3

“Det [emnefunksjonalitet] er liksom ikke noe jeg har savnet, men hvis det skulle komme i det programmet jeg bruker, av og til så ville jeg sikkert brukt det” - P13, 00:54

“Hvis det er en 3-4 stykker, og hvis man skal få noe nyttig ut av samtalen, så kan det jo være [hensiktsmessig med emnefunksjonalitet]” - P14 02:35

“Men samtidig så er det jo slik at, når du går inn en boks der topic er, ja, middag [...], så vet du jo det da, du slipper å bruke kapasitet på å sette deg inn i hva som blir sagt [...] du vet jo hva det dreier seg om” - P14, 03:30

“Kanskje du skulle hatt en funksjon der du flytter en setning som har havnet i feil boks over i en annen boks, for det skjer jo, skjer i alle fall med meg” - P14, 04:10

“Det er ofte når du snakker med en person, som du kanskje ikke ser så ofte, [...], og man har en seriøs ting man skal fikse, samtidig som man snakker mye om løst og fast, så kan det være greit å lage en topic på akkurat den tingen” - P14, 07:58

“Når vi holder på med obliger og sånn, i systemutvikling, da har vi flere emner vi holder på med, og da hadde vært veldig greit å ha det [emnefunksjonalitet] i messengeren” -P15 01:26

“Til profesjonell bruk så er emner veldig bra, men når jeg skal prate med kamerater så tror jeg aldri jeg [ville brukt emnefunksjonalitet]” - P15 02:00

Session 8

“Når det var et vindu jeg var lei av [og lukket], så kom det opp igjen og opp igjen” - P29, 03:20

“Jeg ville brukt det mest i scenarier der jeg snakker med flere personer, som nå i den siste” - P32, 05:30

Session 9

“Det kan være vanskelig å se når nye meldinger kommer opp [når det er mange vinduer]” - P34 03:10

Appendix D – Questionnaires in the original language

Spørreskjema – TopicMessenger (TM) oppgave 1

Du har nå gjennomført en samtale med TopicMessenger. Ranger de følgende uttalelsene. Sett ring rundt et tall – 1 er fullstendig ENIG, 7 er fullstendig UENIG, 4 er verken enig eller uenig.

Utsagn	**ENIG UENIG**						
Jeg ble sjelden avbrutt av den andre i samtalen.	1	2	3	4	5	6	7
Jeg mistet aldri oversikten over samtalen.	1	2	3	4	5	6	7
Jeg klarte alltid å se sammenhengen i det den andre skrev.	1	2	3	4	5	6	7
Funksjonaliteten i TopicMessenger bidro til å gjøre samtalen effektiv.	1	2	3	4	5	6	7
Om jeg hadde hatt tilgang til TopicMessenger til daglig, hadde jeg valgt den til denne type samtaler.	1	2	3	4	5	6	7
TopicMessenger var enkel å bruke.	1	2	3	4	5	6	7

Ditt deltakernummer: ___

Spørreskjema – TopicMessenger (TM) oppgave 2

Du har nå gjennomført en samtale med TopicMessenger. Ranger de følgende uttalelsene. Sett ring rundt et tall – 1 er fullstendig ENIG, 7 er fullstendig UENIG, 4 er verken enig eller uenig.

Utsagn	**ENIG** **UENIG**						
Jeg ble sjelden avbrutt av de andre i samtalen.	1	2	3	4	5	6	7
Jeg mistet aldri oversikten over samtalen.	1	2	3	4	5	6	7
Jeg klarte alltid å se sammenhengen i det de andre skrev.	1	2	3	4	5	6	7
Funksjonaliteten i TopicMessenger bidro til å gjøre samtalen effektiv.	1	2	3	4	5	6	7
Om jeg hadde hatt tilgang til TopicMessenger til daglig, hadde jeg valgt den til denne type samtaler.	1	2	3	4	5	6	7
TopicMessenger var enkel å bruke.	1	2	3	4	5	6	7

Ditt deltakernummer: ___

Spørreskjema – StandardMessenger (SM) oppgave 1

Du har nå gjennomført en samtale med StandardMessenger. Ranger de følgende uttalelsene. Sett ring rundt et tall – 1 er fullstendig ENIG, 7 er fullstendig UENIG, 4 er verken enig eller uenig.

Utsagn	**ENIG** **UENIG**						
Jeg ble sjelden avbrutt av den andre i samtalen.	1	2	3	4	5	6	7
Jeg mistet aldri oversikten over samtalen.	1	2	3	4	5	6	7
Jeg klarte alltid å se sammenhengen i det den andre skrev.	1	2	3	4	5	6	7
Funksjonaliteten i StandardMessenger bidro til å gjøre samtalen effektiv.	1	2	3	4	5	6	7
Om jeg hadde hatt tilgang til StandardMessenger til daglig, hadde jeg valgt den til denne type samtaler.	1	2	3	4	5	6	7
StandardMessenger var enkel å bruke.	1	2	3	4	5	6	7

Ditt deltakernummer: ___

Spørreskjema – StandardMessenger (SM) oppgave 2

Du har nå gjennomført en samtale med StandardMessenger. Ranger de følgende uttalelsene. Sett ring rundt et tall – 1 er fullstendig ENIG, 7 er fullstendig UENIG, 4 er verken enig eller uenig.

Utsagn	**ENIG**						**UENIG**
Jeg ble sjelden avbrutt av de andre i samtalen.	1	2	3	4	5	6	7
Jeg mistet aldri oversikten over samtalen.	1	2	3	4	5	6	7
Jeg klarte alltid å se sammenhengen i det de andre skrev.	1	2	3	4	5	6	7
Funksjonaliteten i StandardMessenger bidro til å gjøre samtalen effektiv.	1	2	3	4	5	6	7
Om jeg hadde hatt tilgang til StandardMessenger til daglig, hadde jeg valgt den til denne type samtaler.	1	2	3	4	5	6	7
StandardMessenger var enkel å bruke.	1	2	3	4	5	6	7

Ditt deltakernummer: ___

Spørreskjema – til slutt

Sammenligning

1 indikerer "Topic Messenger", 7 "Standard Messenger" og 4 er "ingen forskjell".

Utsagn	**TopicMessenger StandardMessenger**						
Med hvilket program ble du oftest avbrutt av samtalepartnerne dine?	1	2	3	4	5	6	7
I hvilket program var det enklest å holde oversikt over samtalen?	1	2	3	4	5	6	7
I hvilket program var det enklest å se sammenhengen i samtalen?	1	2	3	4	5	6	7
I hvilket program var samtalen mest effektiv?	1	2	3	4	5	6	7
Hvilket program ville du brukt til daglig?	1	2	3	4	5	6	7
Hvilket program var enklest å bruke?	1	2	3	4	5	6	7

Ranger uttalelser

Ranger de følgende uttalelsene. Sett ring rundt et tall – 1 er fullstendig ENIG, 7 er fullstendig UENIG.

MERKNAD: Fordi spørsmålene er utarbeidet i henhold til en standard, kan de oppleves som svært like. Forsøk likevel å besvare hvert enkelt spørsmål så oppriktig som mulig.

Utsagn	**ENIG UENIG**						
Jeg vil bruke funksjonalitet for å ha flere emner i samme samtale dersom den blir tilgjengelig i instant messaging-programmet jeg bruker.	1	2	3	4	5	6	7
Dersom funksjonalitet for å ha flere emner i samme samtale var tilgjengelig i mitt vanlige instant messaging-program, ville jeg benyttet meg av det.	1	2	3	4	5	6	7

En slik støtte for emner i mitt instant messenger-program vil gjøre meg mer effektiv i det daglige.	1	2	3	4	5	6	7
Emnefunksjonalitet vil gjøre samtaler mer produktive.	1	2	3	4	5	6	7
Samtaler vil bli mer effektive med støtte for flere emner i samme samtale.	1	2	3	4	5	6	7
Emnefunksjonalitet vil være nyttig å ha i instant messaging-programmet jeg bruker til daglig.	1	2	3	4	5	6	7
Det er klart og lettforståelig å bruke emnefunksjonaliteten i TopicMessenger.	1	2	3	4	5	6	7
Det krever ikke noen stor mental anstrengelse å bruke emnefunksjonaliteten i TopicMessenger.	1	2	3	4	5	6	7
Emnefunksjonaliteten i TopicMessenger er enkel å bruke.	1	2	3	4	5	6	7
Det er lett å forstå hvordan emnefunksjonaliteten i TopicMessenger fungerer.	1	2	3	4	5	6	7

Om deg

Alder: ___ Kjønn: Mann Kvinne *(sett ring rundt)*

Antall år du har brukt instant messaging (ICQ, MSN, Skype, Yahoo! Instant Messaging, Jabber og lignende): ___

Omtrentlig antall instant messaging-samtaler du har i uka: ___

Omtrentlig tid du bruker på instant messaging per uke: Timer: ___ Minutter: ___

Omtrent antall timer du bruker foran datamaskinen i uka: ___

Ditt deltakernummer: ___

Appendix E – Lower-level t-tests

In this appendix, we provide the data from the t-tests conducted on participants having separate roles in the experiment. In other words, the individual participants in these t-tests could not affect each other.

T-tests for pairs and sessions are included for comparison.

Perceived conversation quality, two-way conversations

Role A	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, individual level A	T-test, pairs
I was rarely interrupted...	3.06 (1.73)	2.22 (1.40)	t = -1.974 p = 0.032	t = -2.166 p = 0.022
I never lost overview...	2.7 (1.47)	2.56 (1.46)	t = 0.941 p = 0.180	t = 0.566 p = 0.289
The other party's statements...	2.33 (1.75)	2.5 (1.95)	t = 0.268 p = 0.396	t = -0.458 p =0.326

Role B	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, individual level B	T-test, pairs
I was rarely interrupted...	2.88 (1.83)	2.47 (1.62)	t = -0.749 p = 0.232	t = -2.166 p = 0.022
I never lost overview...	1.88 (1.50)	2.18 (0.93)	t = 0.655 p = 0.261	t = 0.566 p = 0.289
The other party's statements...	2 (1.27)	1.65 (1.11)	t = -0.859 p = 0.201	t = -0.458 p =0.326

Perceived conversation quality, four-way conversations

Role A	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, indiv. A	T-test, sessions
I was rarely interrupted...	5.75* (1.258*)	2.78 (1.093)	t = -5.960 p = 0.005*	t = -10.078 p = 0.001*
I never lost overview...	5.111 (1.167)	3.556 (1.74)	t = -2.485 p = 0.019	t = -1.724 p = 0.061
The other party's statements...	4.667 (1.5)	3.333 (0.866)	t = -2.000 p = 0.040	t = -3.861 p = 0.002

Role B	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, indiv. B	T-test, sessions
I was rarely interrupted...	5.75* (0.5*)	3.25 (1.282)	t = -2.386 p = 0.023*	t = -10.078 p = 0.001*
I never lost overview...	5.375 (1.506)	4.25 (1.832)	t = -1.285 p = 0.120	t = -1.724 p = 0.061
The other party's statements...	4.5 (1.31)	2.375 (1.408)	t = -6.065 p < 0.001	t = -3.861 p = 0.002

Role C	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, indiv. C	T-test, sessions
I was rarely interrupted...	5.5* (0.577)	3.444 (1.81)	t = -2.183 p = 0.059*	t = -10.078 p = 0.001*
I never lost overview...	4 (1.172)	3.778 (1.856)	t = -0.279 p = 0.394	t = -1.724 p = 0.061
The other party's statements...	3.111 (1.364)	3.333 (1.414)	t = 0.406 p = 0.348	t = -3.861 p = 0.002

Role D	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, indiv. D	T-test, sessions
I was rarely interrupted...	6.5* (0.577*)	2.889 (1.616)	t = -3.833 p = 0.016 *	t = -10.078 p = 0.001*
I never lost overview...	3.777 (2.333)	3.333 (2.236)	t = -0.341 p = 0.371	t = -1.724 p = 0.061
The other party's statements...	4 (2.062)	2.778 (1.641)	t = -1.344 p = 0.108	t = -3.861 p = 0.002

User acceptance, two-way conversations

Role A	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, role A	T-test, pairs
The messenger's functionality contributed to making the conversation more efficient	3.72 (1.78)	3.22 (1.63)	t = -0.83 p = 0.21	t = -2.01 p = 0.03
I would use the messenger for conversations like this one	4 (2.07)	4.17 (1.72)	t = 0.27 p = 0.4	t = 0.05 p = 0.48
The messenger was easy to use	1.33 (0.59)	1.56 (0.51)	t = 1.46 p = 0.08	t = 1.19 p = 0.13

Role B	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, role B	T-test, pairs
The messenger's functionality contributed to making the conversation more efficient	3.53 (1.91)	2.47 (1.46)	t = -2.4 p = 0.014	t = -2.01 p = 0.03
I would use the messenger for conversations like this one	3.65 (2)	3.58 (1.42)	t = -0.10 p = 0.46	t = 0.05 p = 0.48
The messenger was easy to use	1.35 (0.79)	1.35 (0.79)	t = 0 p = 0.5	t = 1.19 p = 0.13

User acceptance, four-way conversations

Role A	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, role A	T-test, sessions
The messenger's functionality contributed to making the conversation more efficient	5.78 (1.39)	2.89 (0.93)	t = -4.56 p < 0.001	t = -7.63 p < 0.001
I would use the messenger for conversations like this one	5.33 (1.41)	3.44 (1.24)	t = -2.57 p = 0.02	t = -4.05 p = 0.002
The messenger was easy to use	1.33 (0.71)	2.22 (1.2)	t = 2.1 p = 0.034	t = 1.47 p = 0.09

Role B	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, role B	T-test, sessions
The messenger's functionality contributed to making the conversation more efficient	4.75 (1.75)	3.25 (2.05)	t = -1.27 p = 0.12	t = -7.63 p < 0.001
I would use the messenger for conversations like this one	5.13 (1.37)	3.5 (1.6)	t = -2.73 p = 0.02	t = -4.05 p = 0.002
The messenger was easy to use	2.25 (1.83)	2.125 (1.48)	t = -0.26 p = 0.4	t = 1.47 p = 0.09

Role C	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, role C	T-test, sessions
The messenger's functionality contributed to making the conversation more efficient	5.11 (1.45)	3.44 (1.51)	t = -2.36 p = 0.023	t = -7.63 p < 0.001
I would use the messenger for conversations like this one	5.25 (1.39)	4.44 (1.01)	t = -1.58 p = 0.076	t = -4.05 p = 0.002
The messenger was easy to use	1.67 (0.71)	1.89 (0.93)	t = 0.48 p = 0.32	t = 1.47 p = 0.09

Role D	StandardMessenger Mean (SD)	TopicMessenger Mean (SD)	T-test, role D	T-test, sessions
The messenger's functionality contributed to making the conversation more efficient	5.22 (1.86)	2.77 (0.97)	t = -3.55 p = 0.004	t = -7.63 p < 0.001
I would use the messenger for conversations like this one	4.78 (1.92)	4 (1.5)	t = -0.84 p = 0.212	t = -4.05 p = 0.002
The messenger was easy to use	1.78 (1.09)	2 (1.66)	t = 0.45 p = 0.332	t = 1.47 p = 0.09

Appendix F – Non-parametric test results

In this appendix, Wilcoxon signed-rank test results (one-tailed) are included alongside the t-test results used in the thesis.

Non-parametric test results, two-way conversations

	StandardMessenger	TopicMessenger	Test results	
	Mean (SD)	Mean (SD)	t-test	Wilcoxon
Interruptions per minute	1.527 (0.468)	0.865 (0.198)	t = -3.6 p < 0.001	z = -3.027 p = 0.002
Thread interruptions per minute	0.596 (0.464)	0.129 (0.111)	t = -1.586 p = 0.133	z = -1.437 p = 0.076
Topic interruptions per minute	0.939 (0.560)	0.737 (0.440)	t = -4.096 p < 0.001	z = -3.724 p < 0.001
I was rarely interrupted...	3.03 (1.28)	2.36 (0.87)	t = -2.166 p = 0.022	z = -1.862 p = 0.032
I never lost overview...	2.17 (1.36)	2.33 (0.92)	t = 0.566 p = 0.289	z =-1.782 p = 0.038
The other party's statements...	2.28 (1.45)	2.08 (0.93)	t = -0.458 p =0.326	z = 0 p = 0.5
The messenger's functionality contributed to making the conversation more efficient	3.69 (1.42)	2.83 (1.01)	t = -2.01 p = 0.03	z = -1.856 p = 0.032
I would use the messenger for conversations like this one	3.86 (1.78)	3.89 (1.05)	t = 0.05 p = 0.48	z = 0.087 p = 0.465
The messenger was easy to use	1.33 (0.45)	1.5 (0.49)	t = 1.19 p = 0.13	z = 1.112 p = 0.133

Non-parametric test results, four-way conversations

	StandardMessenger	TopicMessenger	Test results	
	Mean (SD)	Mean (SD)	t-test	Wilcoxon
Interruptions per minute	4.133 (1.579)	1.891 (0.387)	t = -7.358 p < 0.001	z = -2.666 p = 0.004
Thread interruptions per minute	2.872 (1.187)	1.675 (0.555)	t = -4.358 p = 0.002	z = -2.547 p = 0.006
Topic interruptions per minute	1.126 (0.633)	0.215 (0.161)	t = -4.821 p < 0.001	z = -2.547 p = 0.006
I was rarely interrupted...	5.875* (0.25*)	3.167 (0.74)	t = -10.078 p = 0.001*	z = -1.826 p = 0.034
I never lost overview...	4.583 (0.919)	3.694 (0.982)	t = -1.724 p = 0.061	z = -1.424 p = 0.077
The other party's statements...	4.083 (0.673)	2.944 (0.527)	t = -3.861 p = 0.002	z = -2.524 p = 0.006
The messenger's functionality contributed to making the conversation more efficient	5.2 (0.77)	3.09 (0.71)	t = -7.63 p < 0.001	z = -2.670 p = 0.004
I would use the messenger for conversations like this one	5.2 (0.78)	3.09 (0.71)	t = -4.05 p = 0.002	z = -2.524 p = 0.006
The messenger was easy to use	1.74 (0.53)	2.08 (0.68)	t = 1.47 p = 0.09	z = 1.187 p = 0.12

Appendix G – CD with TopicMessenger client and server

The attached CD includes the prototype TopicMessenger client and server. Instructions for running the software are available in the readme.txt file on the CD. Version 1.5 or higher of the Java Runtime is required and can be downloaded from http://java.sun.com/j2se/1.5.0/.